

**Oracle® Workspaces**

Web Services Application Developer's Guide

10g Release 1 (10.1.2.2)

**B28207-01**

March 2006

Oracle Workspaces Web Services Application Developer's Guide, 10g Release 1 (10.1.2.2)

B28207-01

Copyright © 2006, Oracle. All rights reserved.

Primary Author: Raymond Gallardo, Madhubala Mahabaleshwar

Contributor: Shreedhar Patwari, Pradeep Seetharam, Anand Ganapathy, Anand Gomatam, Chaitanya Bhavanasi

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software--Restricted Rights (June 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

Oracle, JD Edwards, PeopleSoft, and Retek are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

---

---

# Contents

<b>Preface</b> .....	v
Intended Audience.....	v
Documentation Accessibility .....	v
Related Documents .....	vi
Conventions .....	vi
<b>1 Overview of Oracle Workspaces Web Services</b>	
Capabilities.....	1-1
Architecture .....	1-2
<b>2 Building Applications with Oracle Workspaces Web Services</b>	
Requirements for Compilation and Execution of Applications that Use Oracle Workspaces Web Services .....	2-1
Classpath of Oracle Workspaces Web Services .....	2-1
Generating Client-Side Stub Classes .....	2-2
Using Apache Ant to Build Client-Side Stub Classes .....	2-2
Compiling Generated Stub Classes .....	2-3
Location of WSDL Files .....	2-3
General Procedure to Build Applications with Oracle Workspaces Web Services .....	2-3
<b>3 Authentication Service</b>	
Example: Retrieve Authentication Cookie .....	3-1
<b>4 Service-to-Service Authentication Service</b>	
Setting Oracle Workspaces Environment Variables .....	4-1
Registering Partner Applications with Oracle Internet Directory .....	4-1
Setting Oracle-Specific and Digest Authentication HTTP Headers, and Invoking the S2S Authentication Service .....	4-2
Retrieving Authentication Cookie without S2S Authentication Service.....	4-3
<b>5 Home Service</b>	
Example: Create New Workspace.....	5-1

<b>6</b>	<b>Users Service</b>	
	Example: Retrieve Users.....	6-1
<b>7</b>	<b>Workspaces Service</b>	
	Example: Add Members and Resources to a Workspace.....	7-1
<b>8</b>	<b>Library Service</b>	
	Example: Create Folders in and Retrieve Contents of Library.....	8-1
<b>9</b>	<b>Meetings Service</b>	
	Example: Create Meeting.....	9-1
<b>10</b>	<b>Tasks Service</b>	
	Example: Create and Search for Tasks.....	10-1
<b>11</b>	<b>Announcements Service</b>	
	Example: Create Announcement.....	11-1
<b>12</b>	<b>Discussion Service</b>	
	Example: Create Board, Post Message, and Post Reply.....	12-1
<b>13</b>	<b>Inbox Service</b>	
	Example: Post Message to Inbox.....	13-1
<b>14</b>	<b>Views Service</b>	
	Example: Create View, Add Items to View, and Search for Views that Contain Item.....	14-1
<b>15</b>	<b>Attachments Service</b>	
	Example: Create Attachment.....	15-1
<b>16</b>	<b>Template Service</b>	
	Example: Retrieve Templates.....	16-1
<b>17</b>	<b>Administration Service</b>	
	Example: Grant Roles to Members.....	17-1

**Index**

---

---

# Preface

This preface contains the following topics:

- [Intended Audience](#)
- [Documentation Accessibility](#)
- [Related Documents](#)
- [Conventions](#)

## Intended Audience

*Oracle Workspaces Web Services Application Developer's Guide* is intended for any programmers and developers who intend to use the Oracle Workspaces Web services.

## Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at

<http://www.oracle.com/accessibility/>

### **Accessibility of Code Examples in Documentation**

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

### **Accessibility of Links to External Web Sites in Documentation**

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

### TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

## Related Documents

For more information, see the following manuals in the Oracle Collaboration Suite documentation set:

- *Oracle Workspaces Web Services Java API Reference*
- *Oracle Workspaces Application Developer's Guide*
- *Oracle Workspaces Java API Reference*

## Conventions

The following conventions are also used in this manual:

Convention	Meaning
. . .	Vertical ellipsis points in an example mean that information not directly related to the example has been omitted.
...	Horizontal ellipsis points in statements or commands mean that parts of the statement or command not directly related to the example have been omitted
<b>boldface text</b>	Boldface type in text indicates a term defined in the text, the glossary, or in both locations.
<>	Angle brackets enclose user-supplied names.
[ ]	Brackets enclose optional clauses from which you can choose one or none.

---

---

# Overview of Oracle Workspaces Web Services

Oracle Workspaces is an extremely flexible framework that organizes and links Oracle Collaboration Suite components and their data.

This chapter covers the following topics:

- [Capabilities](#)
- [Architecture](#)

## Capabilities

The Oracle Workspaces Web Services is a layer on top of Oracle Workspaces SDK. It exposes operations supported by Oracle Workspaces SDK as Web services. These operations have been categorized into the following services:

- [Authentication Service](#): Provides methods for the user's login and logout operations.
- [Administration Service](#): Provides methods to perform administrative operations on workspaces.
- [Announcements Service](#): Provides methods to operate on announcements in a workspace.
- [Attachments Service](#): Provides methods to operate on attachments within a workspace.
- [Discussion Service](#): Provides methods to operate on discussions in a workspace.
- [Home Service](#): Provides methods to manage workspaces life-cycle operations.
- [Inbox Service](#): Provides methods to operate on inbox resource within a workspace.
- [Library Service](#): Provides methods to operate on content within a workspace.
- [Meetings Service](#): Provides methods to operate on meetings in a workspace.
- [Service-to-Service Authentication Service](#): Provides authentication mechanism for the other applications to invoke operations on Oracle Workspaces.
- [Tasks Service](#): Provides methods to operate on Tasks within a workspace.
- [Template Service](#): Provides methods to perform operations on workspace templates.
- [Users Service](#): Provides methods to operate on properties of users of workspaces.
- [Views Service](#): Provides methods to operate on views within a workspace.

- [Workspaces Service](#): Provides methods to perform operations on a workspace.

---

---

**See Also:** For more information about Oracle Workspaces SDK, see *Oracle Workspaces Application Developer's Guide*.

---

---

## Architecture

Oracle Workspaces Web Services follows the Java API for XML-based RPC (JAX-RPC) specification.

---

---

**See Also:** For more information about JAX-RPC, visit <http://java.sun.com/webservices/jaxrpc/>.

For information about the architecture of Oracle Workspaces, see "Architecture of Oracle Workspaces" in *Oracle Workspaces Application Developer's Guide*.

For more information about Web services specifications, visit the following links:

- Web Services Activity, <http://www.w3.org/2002/ws/>
  - Web Services Description Language (WSDL) 1.1, <http://www.w3.org/TR/wSDL>
  - Simple Object Access Protocol (SOAP) 1.1, <http://www.w3.org/TR/soap/>
- 
-

---

---

## Building Applications with Oracle Workspaces Web Services

This chapter describes the following requirements and tasks involved in building applications with Oracle Workspaces Web Services:

- [Requirements for Compilation and Execution of Applications that Use Oracle Workspaces Web Services](#)
  - [Classpath of Oracle Workspaces Web Services](#)
  - [Generating Client-Side Stub Classes](#)
    - \* [Using Apache Ant to Build Client-Side Stub Classes](#)
  - [Compiling Generated Stub Classes](#)
- [Location of WSDL Files](#)
- [General Procedure to Build Applications with Oracle Workspaces Web Services](#)

### Requirements for Compilation and Execution of Applications that Use Oracle Workspaces Web Services

You will need a SOAP engine that can generate client stub classes from WSDL files, such as Apache Axis, Oracle JDeveloper, or Oracle Application Server SOAP Management Utilities and Scripts.

This guide uses Apache Axis as its SOAP engine and Apache Xerces Java Parser. The version of Apache Axis should be at least 1.2.0.

---

---

**See Also:** For more information about Apache Axis, visit <http://ws.apache.org/axis/>.

For more information about Apache Xerces, visit <http://xerces.apache.org/xerces-j/>.

---

---

### Classpath of Oracle Workspaces Web Services

On the machine you wish to execute your Oracle Workspaces Web services client application, set the classpath to include the following libraries and files:

- Library .jar files shipped with Apache Axis
- Libraries that Apache Axis require. Apache Axis requires a JAXP-1.1 compliant XML parser such as Apache Xerces (the samples in this guide use Xerces).

## Generating Client-Side Stub Classes

Before using any Oracle Workspaces Web Services, you must create client-side stub classes from WSDL files hosted on the Oracle Workspaces server .

Generate client-side stub classes of a particular Web service by using the Apache Axis WSDL2Java tool:

```
java -cp <AXISPATH> org.apache.axis.wsdl.WSDL2Java
  --deployScope Session
  --NStoPkg <package-to-namespace-mapping>
  <path-to-wsdl-file>
```

<AXISPATH> is the classpath that contains the Apache Axis library jar files.

<package-to-namespace-mapping> is the package to namespace mapping. The following is an example of a mapping:

```
"http://xmlns.oracle.com/workspaces/ws"="oracle.workspaces.ws"
```

<path-to-wsdl-file> is the URL where the WSDL files are hosted. This path is in the following form:

```
http://<host>:<port>/ocw/services/<service-name>?wsdl
```

- <host>:<port> is the host and port where the WSDL files are hosted.
- <service-name> is the name of the Web service for which you wish to generate the client-side bindings

For example, if your WSDL files are hosted in your local OC4J instance at port 8888 on machine www.example.com and you wish to generate the stubs for AuthenticationService, the URL would be the following:

```
http://www.example.com:8888/ocw/ws/AuthenticationService?wsdl
```

## Using Apache Ant to Build Client-Side Stub Classes

Alternatively, you may use Apache Ant and Axis Ant tasks to generate client-side stub classes. The following is an example of using the <axis-wsdl2java> task to generate the client-side stubs for AuthenticationService:

```
<axis-wsdl2java
  output="{GEN_HOME}/src"
  testcase="true"
  verbose="true"
  deployscope="Session"
  url="{WS_HOST}/ocw/services/AuthenticationService?wsdl">
  <mapping
    namespace="http://xmlns.oracle.com/workspaces/ws"
    package="oracle.workspaces.ws" />
</axis-wsdl2java>
```

GEN\_HOME is the directory in which the client-side stub .java files will be generated

WS\_HOST is the URL where the WSDL files are hosted.

---

---

**See Also:** For more information about Apache Ant, visit  
<http://ant.apache.org/>.

---

---

## Compiling Generated Stub Classes

Compile the generated stub classes and package them into a .jar file. Ensure that you add this .jar file to the classpath of your Oracle Workspaces Web services client application.

## Location of WSDL Files

A list of all available Oracle Workspaces Web services is available at the following URL:

```
http://<host>:<port>/ocw/ws
```

The WSDL file for a particular Web service is available at the following URL:

```
http://<host>:<port>/ocw/services/<name-of-Web-service>?wsdl
```

*<name-of-Web-service>* is the name of the Web service.

## General Procedure to Build Applications with Oracle Workspaces Web Services

The following steps are a general outline you may follow to create Oracle Workspaces Web Services client applications:

1. Use [Authentication Service](#) or [Service-to-Service Authentication Service](#) to retrieve an authentication cookie. This authentication cookie is required to invoke any Oracle Workspaces Web service.
2. Most Oracle Workspaces Web service operations require a workspace. Use [Home Service](#) to create or retrieve a workspace.
3. Retrieve users of Oracle Workspaces (which are users of Oracle Collaboration Suite that have been registered in Oracle Internet Directory) with [Users Service](#).
4. Make a user a member of a workspace with [Workspaces Service](#). You may also add additional resources to a workspace with this service.
5. Work with workspace items with the following services:
  - [Library Service](#)
  - [Meetings Service](#)
  - [Tasks Service](#)
  - [Announcements Service](#)
  - [Discussion Service](#)
  - [Inbox Service](#)
  - [Views Service](#)
  - [Attachments Service](#)
6. Retrieve, create, store, and delete workspace templates with [Template Service](#).
7. Grant or revoke administrative roles to users or configure Oracle Workspaces with [Administration Service](#).



---

---

## Authentication Service

The Authentication service provides an authentication mechanism for users to access Oracle Workspaces Web services.

The user supplies his or her login credentials. The Authentication service Web service validates them by connecting to Oracle Internet Directory. An authentication cookie representing the user session is returned. Pass this cookie to all other Oracle Workspaces Web services in order to invoke them. As a result, the user will not be required to supply his credentials again until the cookies expires or the user explicitly logs out.

See the code samples in chapters about specific services such as [Chapter 5, "Home Service"](#) for examples of how to use the authentication cookie to invoke Oracle Workspaces Web services.

### Example: Retrieve Authentication Cookie

The following code sample retrieves an authentication cookie:

**Example 3-1 AuthenticationSample.java**

```
package oracle.sample.workspaces.ws;

import java.rmi.RemoteException;
import javax.xml.rpc.ServiceException;
import oracle.workspaces.ws.AuthenticationService;
import oracle.workspaces.ws.AuthenticationServiceServiceLocator;
import oracle.workspaces.ws.AuthenticationServiceSoapBindingStub;
import oracle.workspaces.ws.S2SAuthenticationServiceSoapBindingStub;
import oracle.workspaces.ws.exceptions.CwWSEException;
import org.apache.axis.transport.http.HTTPConstants;

public class AuthenticationSample
{
    public static AuthenticationService
        configureAuthenticationService(
            String username,
            String password) throws ServiceException, RemoteException, CwWSEException
    {
        System.out.println("Logging on to Oracle Workspaces Web services " +
            "using AuthenticationService");

        // Initialize the service locator for the Authentication service.
        // The service locator contains the Web service endpoint URL
        // in the form of
        // http://<midtier instance name>:<port>/
```

```
//    ocw/services/AuthenticationService

AuthenticationServiceServiceLocator assl =
    new AuthenticationServiceServiceLocator();

// Retrieve a reference to the Web service's remote interface
// from the service locator

AuthenticationService as = assl.getAuthenticationService();

// Indicate the client's willingness to participate in the session.
// Unless it is set to true, the server assumes that client is not
// participating in the session.
//
// As HTTP is stateless, each Web service invocation will be
// different and the user's state will not be
// maintained across different Web service invocations.

((AuthenticationServiceSoapBindingStub)as).
    setMaintainSession(true);

// Invoke the login method to authenticate the user.
// User nickname and password are sent as plain text.

as.login(username, password);

return as;

}

public static String getAuthenticationCookie
    (AuthenticationService as)
    throws CwWSEException, ServiceException, RemoteException
{
    // Once the authentication is successful, an HTTP session is
    // established between the Web services server and the user.
    //
    // A J2EE session is established between the Web services server
    // and the Oracle Workspaces back-end. A cookie identifying the
    // user's HTTP session is placed into the servlet application context.
    //
    // This cookie is to be retrieved and supplied on each
    // subsequent Web services invocation.

    String cookie = (String)((AuthenticationServiceSoapBindingStub)as).
        _getCall().getMessageContext().
        getProperty(HTTPConstants.HEADER_COOKIE);

    return cookie;
}

public static void main(String[] args)
{
    try {

        // Get authentication cookie

        AuthenticationService myAuthenticationService =
```

```
AuthenticationSample.configureAuthenticationService(  
    "orcladmin",  
    "welcome1");  
  
String authCookie = AuthenticationSample.getAuthenticationCookie  
    (myAuthenticationService);  
  
System.out.println("Retrieved authentication cookie : " + authCookie);  
  
// Invoke the logout method to destroy the user credentials  
// stored in the HTTP session.  
//  
// On logout invocation, the HTTP session is invalidated and  
// the cookie is destroyed.  
//  
// The user's state is lost and hence the user will not be able  
// to invoke any other Web services methods  
// until he or she logs in again.  
  
myAuthenticationService.logout();  
  
} catch (CwWSEException cwse)  
{  
    System.out.println("CwWSEException caught: " + cwse.getMessage());  
} catch (Exception e)  
{  
    System.out.println("Exception caught: " + e.toString());  
}  
}  
}
```



---

## Service-to-Service Authentication Service

The service-to-service (S2S) authentication framework allows a trusted partner application to establish user sessions with a trusting provider application on behalf of its users. As a result, the partner application does not require the credentials of the end-user using it. Instead, the partner application itself supplies its credentials to establish a user session with the provider application.

The S2S Authentication service allows an application developed with Oracle Workspaces Web services (the partner application) to establish user sessions with Oracle Workspaces (the provider application). The partner application does not require the credentials of the end-user using it.

Follow these steps to integrate the S2S authentication framework with your partner application:

1. [Setting Oracle Workspaces Environment Variables](#)
2. [Registering Partner Applications with Oracle Internet Directory](#)
3. [Setting Oracle-Specific and Digest Authentication HTTP Headers, and Invoking the S2S Authentication Service](#)

In some cases, you may not be able to use the S2S Authentication Web service. See [Retrieving Authentication Cookie without S2S Authentication Service](#) for more information.

### Setting Oracle Workspaces Environment Variables

The environment property "oracle.workspaces.ws.s2sEnabled" must be set to true. By default, this property is set to true. You may set this property in Oracle Enterprise Manager Application Server Control. Go to the Applications tier, **OC4J\_OCSCClient** system component, **workspaces** application, **workspaces\_ws** Web module, and follow the link to **Environment** to access Oracle Workspaces environment entries.

### Registering Partner Applications with Oracle Internet Directory

The following steps describe how to register a partner application with the S2S authentication framework:

1. Create the following entry in Oracle Internet Directory:

```
dn: cn=MyApplicationProductName,cn=Products, cn=OracleContext
objectClass: orclContainer
objectClass: top
```

*MyApplicationProductName* is the product name (or category) of your application.

2. Create the following entry in Oracle Internet Directory:

```
dn: orclApplicationCommonName=MyAppName,  
cn=MyApplicationProductName,cn=Products,  
cn=OracleContext  
objectClass: orclApplicationEntity  
objectClass: top  
orclApplicationCommonName: MyAppName  
userpassword: ApplicationPassword
```

*MyAppName* is the name of your application. *ApplicationPassword* is the password to access your application.

3. To the *MyAppName* entity you have just added, find the property `orcltrustedapplicationgroup`. Set the value of this property to the name of the Trusted Applications group, which should be the following:

```
cn=trusted applications,cn=groups,cn=oraclecontext
```

4. Find the following entry in Oracle Internet Directory:

```
dn: cn=Trusted Applications,cn=Groups,cn=OracleContext
```

To the entry that you have just found, add the following line to `uniquemember` (all in lower case, line breaks added for clarity):

```
orclapplicationcommonname=myappname,  
cn=myapplicationproductname,cn=products,cn=oraclecontext
```

Use Oracle Directory Manager or the command-line tools provided by Oracle Internet Directory to add and modify entries in Oracle Internet Directory. For more information about these tools, see Chapter 4, "Directory Administration Tools" in Oracle Internet Directory Administrator's Guide.

## Setting Oracle-Specific and Digest Authentication HTTP Headers, and Invoking the S2S Authentication Service

Your SOAP client must support HTTP digest authentication in order to use the S2S Authentication service. See [Retrieving Authentication Cookie without S2S Authentication Service](#) if you are using a client that does not support digest authentication such as Apache Axis.

If your SOAP client supports digest authentication, follow these steps to invoke the S2S Authentication service:

1. Obtain the `S2SAuthenticationService`.
2. Indicate the client's willingness to participate in a session by calling the method `setMaintainSession(true)`.
3. Set the digest authentication HTTP headers that identify the partner service.
4. Set the Oracle-specific header `ORA_S2S_PROXY_USER` to the value of a nickname of an Oracle Collaboration Suite user. The S2S authentication framework will use this user to authenticate the partner application.
5. Invoke the `login()` method to authenticate the user.

## Retrieving Authentication Cookie without S2S Authentication Service

Some Web services clients do not allow access to the underlying HTTP transport mechanism and thus prevent you from setting any HTTP headers. These clients are thus incapable of using the S2S Authentication service.

Oracle Workspaces Web services provides you with `S2SAuthenticationServlet`, a servlet from which an application can retrieve an authentication cookie without the credentials of an end-user. This cookie may then be used to invoke any other service from Oracle Workspaces Web services in the same way that the cookie retrieved from Authentication services is used.

The following code demonstrates how to use the `S2SAuthenticationServlet`. It calls the method `getSessionCookies()`, which takes the following parameters:

- The nickname of the Oracle Collaboration Suite user with which the S2S authentication framework will authenticate the partner application
- The distinguished name of the partner application, which has been registered in Oracle Internet Directory
- The partner application's password, which also has been registered in Oracle Internet Directory
- The realm of the partner application
- The URL from which the `S2SAuthenticationServlet` is deployed

If you wish to use this code as-is, make sure that you have registered the string specified by the variable `appName` with Oracle Internet Directory as described previously.

This sample requires the `HTTPClient` library as well as the `JavaMail` API `mail.jar` in your classpath.

---



---

**See Also:** For more information about the `HTTPClient` package, see the `HTTPClient` API Reference Javadoc in the Oracle Application Server Documentation Library.

For more information about `JavaMail`, visit <http://java.sun.com/products/javamail/>

---



---

### Example 4-1 `S2SClientSample.java`

```
package oracle.sample.workspaces.ws;

import HTTPClient.Cookie;
import HTTPClient.CookieModule;
import HTTPClient.HTTPConnection;
import HTTPClient.HTTPResponse;
import HTTPClient.ModuleException;
import HTTPClient.NVPair;
import HTTPClient.ProtocolNotSuppException;
import java.io.IOException;
import java.io.UnsupportedEncodingException;
import java.net.MalformedURLException;
import java.net.URL;
import javax.mail.internet.MimeUtility;
import oracle.workspaces.ws.exceptions.CwWSException;

public class S2SClientSample
```

```
{
    public static String getSessionCookies(
        String szUsername,
        String szApplicationName,
        String szApplicationPassword,
        String szApplicationRealm,
        URL szApplicationURL) throws CwWSEException
    {

        // Clear the default CookiePolicyHandler
        // so that it accepts all cookies

        CookieModule.setCookiePolicyHandler(null);

        // The application dn should be in lower case

        szApplicationName = szApplicationName.toLowerCase();

        int status = 0;
        Cookie[] cookies = null;
        String cookieValues = "";
        Object context = new Object();

        try
        {
            try
            {
                // Get a HTTP connection for this context

                System.out.println("Connecting..");
                HTTPConnection connection =
                    new HTTPConnection(szApplicationURL);
                connection.setContext(context);
                connection.setAllowUserInteraction(false);

                // Set the digest authorization information

                System.out.println("Adding Digest Auth..");

                connection.addDigestAuthorization(
                    szApplicationRealm,
                    szApplicationName,
                    szApplicationPassword);

                // Mime encode the username

                System.out.println("Encoding username.." + szUsername);

                String encodedUsername = MimeUtility.encodeWord(szUsername);

                // Set the ORA_S2S_PROXY_USER header

                NVPair[] headers = new NVPair[] {
                    new NVPair("ORA_S2S_PROXY_USER", encodedUsername)
                };

                // Send a request to the S2S servlet

                System.out.println("Retrieving response...");
            }
        }
    }
}
```

```
    HTTPResponse s2sResponse =
        connection.Get(
            szApplicationURL.getFile(), (NVPair[])null, headers);

    // Get the request status and cookies

    status = s2sResponse.getStatusCode();

    System.out.println("Status from response: " + status);
    System.out.println("Reason line from response: " +
        s2sResponse.getReasonLine());

    // Close out the sockets
    s2sResponse.getInputStream().close();
    connection.stop();
}

catch(UnsupportedEncodingException uee)
{
    System.out.println("UnsupportedEncodingException caught: " +
        uee.toString());
    CwWSEException tdex = new CwWSEException();
    tdex.setErrorMessage(uee.getLocalizedMessage());
    throw tdex;
}

catch(ProtocolNotSuppException pnse)
{
    System.out.println("ProtocolNotSuppException caught: " +
        pnse.toString());
    CwWSEException tdex = new CwWSEException();
    tdex.setErrorMessage(pnse.getLocalizedMessage());
    throw tdex;
}

catch(IOException ioe)
{
    System.out.println("IOException caught: " +
        ioe.toString());
    CwWSEException tdex = new CwWSEException();
    tdex.setErrorMessage(ioe.getLocalizedMessage());
    throw tdex;
}

catch(ModuleException me)
{
    System.out.println("ModuleException caught: " + me.toString());
    CwWSEException tdex = new CwWSEException();
    tdex.setErrorMessage(me.getLocalizedMessage());
    throw tdex;
}

// If the request succeeded then get the cookie values

if (status == 200) {

    System.out.println("Cookies from context: " + cookies == null);

    cookies = CookieModule.listAllCookies(context);
```

```
        for(int i = 0 ; (cookies!=null) && (i<cookies.length) ; i++)
        {
            String value = cookies[i].getName() + "=" + cookies[i].getValue();
            if (i == 0)
            {
                cookieValues = value;
            }
            else
            {
                cookieValues = cookieValues + ";" + value;
            }
        }
    }
    // else throw an exception
    else
    {
        CwWSEException tdex = new CwWSEException();
        tdex.setErrorMessage("Response status returned was :" + status);
        throw tdex;
    }
}
finally
{
    CookieModule.discardAllCookies(context) ;
}
return cookieValues;
}
public static void main(String[] args)
{
    // Provide the application distinguished name that you
    // have registered in Oracle Internet Directory

    String appName = "orclApplicationCommonName=MyAppName," +
        "cn=MyApplicationProductName," +
        "cn=Products,cn=OracleContext";

    // Provide the application password attribute. This is the value of
    // userpassword of the application in Oracle Internet Directory.

    String appPwd = "welcome1";

    // Supply the URL where the S2SAuthenticationServlet is
    // deployed

    String appURL = "http://www.example.com:7777" +
        "/ocw/s2s/S2SAuthenticationServlet";

    // Provide the user nickname on behalf of whom the caller application
    // is proxying. The user should be a valid Oracle Collaboration
    // Suite user.

    // The user orcladmin is used in this example

    String sessionCookies = "";

    try {

        // Call the getSessionCookies method in this application

        sessionCookies = getSessionCookies("orcladmin", appName, appPwd,
```

```
        "", new URL(appURL));
    } catch (MalformedURLException mue)
    {
        System.out.println("MalformedURLException caught: " + mue.toString());
    }
    catch(CwWSEException cwse)
    {
        System.out.println("CwWSEException caught: " + cwse.getMessage());
    }

    // Use the session cookies that you have retrieved for
    // future calls to Oracle Workspaces Web services

    System.out.println("Session cookies:");
    System.out.println(sessionCookies);
}
}
```



---

---

## Home Service

The Home service allows you to perform workspace life cycle operations such as creating, listing, and deleting workspaces and retrieving workspaces by ID or path. The Home service also allows you to search for content within a workspace.

### Example: Create New Workspace

The following sample will create a new workspace. If the workspace already exists it will try to retrieve it.

#### *Example 5-1 HomeServiceSample.java*

```
package oracle.sample.workspaces.ws;

import java.rmi.RemoteException;
import javax.xml.rpc.ServiceException;
import oracle.workspaces.ws.AuthenticationService;
import oracle.workspaces.ws.HomeService;
import oracle.workspaces.ws.HomeServiceServiceLocator;
import oracle.workspaces.ws.HomeServiceSoapBindingStub;
import oracle.workspaces.ws.beans.CreateWorkspaceResponseItem;
import oracle.workspaces.ws.beans.WorkspaceDefinition;
import oracle.workspaces.ws.beans.WorkspaceItem;
import oracle.workspaces.ws.exceptions.CwWSEException;
import org.apache.axis.transport.http.HTTPConstants;

public class HomeServiceSample
{
    public static HomeService configureHomeService(
        String szAuthCookie) throws ServiceException
    {
        // Once authenticated, the user can invoke any Oracle
        // Workspaces Web service until he or she invokes the logout method.
        //
        // The cookie retrieved from the Authentication service
        // is to be set on every client stub that invokes operations on
        // the Web services server.
        //
        // Initialize the service locator for the Home service.
        // The service location contains the Web services endpoint URL
        // for the category service.

        HomeServiceServiceLocator hssl = new HomeServiceServiceLocator();

        // Retrieve a reference to the remote Home service.
```

```
HomeService hs = hssl.getHomeService();

// Indicate the client's willingness to participate in the session.

((HomeServiceSoapBindingStub)hs).setMaintainSession(true);

// Set the cookie before invoking the operation on the stub.
// Setting this cookie indicates that the client's HTTP session
// credentials have been sent to the server.
//
// Sessions are managed through cookies.

((javax.xml.rpc.Stub)hs).
    _setProperty(HTTPConstants.HEADER_COOKIE, szAuthCookie);

// Now you may use the reference to the remote Home service to
// invoke Web services methods to perform various operations.

return hs;
}

public static WorkspaceItem createOrRetrieveWorkspace(
    HomeService szHomeService,
    String szWorkspaceName,
    String szWorkspaceDisplayName,
    String szWorkspaceDescription,
    String szDefaultMemberRoleType,
    String szMemberAccessSetting)
    throws CwWSEException, RemoteException
{
    WorkspaceItem wspc = null;

    // Create workspace definition object

    WorkspaceDefinition wspcDef = new WorkspaceDefinition();
    wspcDef.setName(szWorkspaceName);
    wspcDef.setDisplayName(szWorkspaceDisplayName);
    wspcDef.setDescription(szWorkspaceDescription);
    wspcDef.setDefaultMemberRoleType(szDefaultMemberRoleType);
    wspcDef.setMemberAccessSetting(szMemberAccessSetting);
    wspcDef.setPubliclyListed(true);
    wspcDef.setAllowPartialSuccess(true);

    try {

        // Invoke method on home service to create a new workspace

        CreateWorkspaceResponseItem resp =
            szHomeService.createWorkspace(wspcDef);
        wspc = resp.getWorkspaceItem();
    }
    catch(CwWSEException e)
    {
        // An exception will be thrown if the
        // workspace already exists

        System.out.println(
            "CwWSEException caught: " + e.getMessage());
    }
}
```

```
try {

    // Try to retrieve the workspace if
    // it exists

    System.out.println("Trying to retrieve " + "/" +
        szWorkspaceName.toUpperCase());

    wspc = szHomeService.getWorkspaceByPath(
        "/" + szWorkspaceName.toUpperCase());
} catch (CwWSEException e2)
{
    System.out.println("CwWSEException caught trying to retrieve " +
        "workspace: " + e2.getMessage());
    throw e2;
}
}
return wspc;
}

public static void main(String[] args)
{
    try {

        // Get authentication cookie

        AuthenticationService myAuthenticationService =
            AuthenticationSample.configureAuthenticationService(
                "orcladmin",
                "welcome1");

        String authCookie = AuthenticationSample.getAuthenticationCookie
            (myAuthenticationService);

        // Retrieve HomeService and set authentication cookie
        HomeService myHomeService =
            HomeServiceSample.configureHomeService(authCookie);

        // Create new workspace
        WorkspaceItem myWorkspace =
            HomeServiceSample.createOrRetrieveWorkspace(
                myHomeService,
                "NewWorkspace",
                "My New Workspace",
                "Description of My New Workspace",
                "WRITER",
                "ENABLED");

        // Display properties of newly created workspace
        System.out.println("Details of newly created " +
            "(or retrieved) workspace:");
        System.out.println("Workspace Id: " +
            myWorkspace.getWorkspaceId());
        System.out.println("ContactEmailAddress: " +
            myWorkspace.getContactEmailAddress());
        System.out.println("Workspace Name: " +
            myWorkspace.getName());
        System.out.println("Workspace Owner: " +
```

```
        myWorkspace.getOwner());
    System.out.println("Publicly Listed: " +
        myWorkspace.isPubliclyListed());
    System.out.println("Path: " +
        myWorkspace.getPath());

    // Invoke the logout method to destroy the user credentials
    // stored in the HTTP session.
    //
    // On logout invocation, the HTTP session is invalidated and
    // the cookie is destroyed.
    //
    // The user's state is lost and hence the user will not be able
    // to invoke any other Web services methods
    // until he or she logs in again.

    myAuthenticationService.logout();

} catch (CwWSEException cwse)
{
    System.out.println("CwWSEException caught: " + cwse.getMessage());
} catch (Exception e)
{
    System.out.println("Exception caught: " + e.toString());
}
}
}
```

---

---

## Users Service

The Users service allows you to retrieve users by email, nickname, or Oracle Internet Directory group and reading and modifying user preferences.

### Example: Retrieve Users

The following sample retrieves users from a specified group that has been defined in Oracle Internet Directory. Before executing the sample, either add users to the group as specified in the sample or specify your own group in Oracle Internet Directory and populate it with users.

#### *Example 6-1 UsersServiceSample.java*

```
package oracle.sample.workspaces.ws;

import java.rmi.RemoteException;
import javax.xml.rpc.ServiceException;
import oracle.workspaces.ws.AuthenticationService;
import oracle.workspaces.ws.UsersService;
import oracle.workspaces.ws.UsersServiceServiceLocator;
import oracle.workspaces.ws.UsersServiceSoapBindingStub;
import oracle.workspaces.ws.beans.User;
import oracle.workspaces.ws.exceptions.CwWSEException;
import org.apache.axis.transport.http.HTTPConstants;

public class UsersServiceSample
{
    public static UsersService configureUsersService(
        String szAuthCookie)
        throws ServiceException
    {
        UsersServiceServiceLocator ussl = new UsersServiceServiceLocator();
        UsersService uService = ussl.getUsersService();
        ((UsersServiceSoapBindingStub)uService).setMaintainSession(true);
        ((javax.xml.rpc.Stub)uService).
            _setProperty(HTTPConstants.HEADER_COOKIE, szAuthCookie);
        return uService;
    }

    public static User[] getUsersFromGroupDN(
        UsersService szUsersService,
        String szGroupDN) throws CwWSEException, RemoteException
    {
        oracle.workspaces.ws.beans.User[] value = null;
        value = szUsersService.getMembersByGroupDN(szGroupDN, null);
    }
}
```

```
// Output users retrieved from Oracle Internet Directory

if (value != null) {
    for(int i=0; i<value.length; i++) {
        System.out.println("User" + i + ":" + value[i].getNickname());
    }
}
return value;
}

public static void main(String[] args)
{
    try {

// Get authentication cookie

        AuthenticationService myAuthenticationService =
            AuthenticationSample.configureAuthenticationService(
                "orcladmin",
                "welcome1");

        String authCookie = AuthenticationSample.getAuthenticationCookie
            (myAuthenticationService);

        System.out.println("Retrieved authentication cookie : " + authCookie);

// Get UsersService and set authentication cookie
        UsersService myUsersService =
            UsersServiceSample.configureUsersService(authCookie);

// Output users from specified DN

        UsersServiceSample.getUsersFromGroupDN(
            myUsersService,
            "cn=raymond_group,cn=Groups,dc=us,dc=oracle,dc=com");

        myAuthenticationService.logout();

    } catch (CwWSEException cwse)
    {
        System.out.println("CwWSEException caught: " + cwse.getMessage());
    } catch (Exception e)
    {
        System.out.println("Exception caught: " + e.toString());
    }
}
}
```

---

---

## Workspaces Service

The Workspaces service allows you to create, update, and remove resources and manage users of a workspace.

### Example: Add Members and Resources to a Workspace

The following sample retrieves users from a specified group in Oracle Internet Directory and adds them as members of a workspace. Each newly added member is given writer role. Afterwards, the sample adds a calendar, library, discussions, and inbox resource to the same workspace. An exception is thrown if any of these resources already exist.

#### *Example 7-1 WorkspacesServiceSample.java*

```
package oracle.sample.workspaces.ws;

import java.rmi.RemoteException;
import javax.xml.rpc.ServiceException;
import oracle.workspaces.ws.AuthenticationService;
import oracle.workspaces.ws.HomeService;
import oracle.workspaces.ws.UsersService;
import oracle.workspaces.ws.WorkspacesService;
import oracle.workspaces.ws.WorkspacesServiceServiceLocator;
import oracle.workspaces.ws.WorkspacesServiceSoapBindingStub;
import oracle.workspaces.ws.beans.CalendarResourceDefinition;
import oracle.workspaces.ws.beans.DiscussionResourceDefinition;
import oracle.workspaces.ws.beans.InboxResourceDefinition;
import oracle.workspaces.ws.beans.LibraryResourceDefinition;
import oracle.workspaces.ws.beans.Member;
import oracle.workspaces.ws.beans.User;
import oracle.workspaces.ws.beans.WorkspaceItem;
import oracle.workspaces.ws.exceptions.CwWSEException;
import org.apache.axis.transport.http.HTTPConstants;

public class WorkspacesServiceSample
{
    public static WorkspacesService configureWorkspacesService(
        String szAuthCookie) throws ServiceException
    {
        WorkspacesServiceServiceLocator wssl =
            new WorkspacesServiceServiceLocator();
        WorkspacesService wService = wssl.getWorkspacesService();
        ((WorkspacesServiceSoapBindingStub)wService).setMaintainSession(true);
        ((javax.xml.rpc.Stub)wService).
            _setProperty(HTTPConstants.HEADER_COOKIE, szAuthCookie);
    }
}
```

```
        return wService;
    }

    public static void createResources(
        WorkspacesService szWorkspacesService,
        String szWorkspaceID)
    {
        // This method attempts to add a calendar, discussions,
        // library, and inbox resource in the specified workspace.
        // An exception is thrown if any of these resources exist
        // in the workspace.

        System.out.println("Adding calendar resource");
        CalendarResourceDefinition rDef = new CalendarResourceDefinition();
        rDef.setDescription("Calendar resource");
        rDef.setName("Calendar1");

        try {

            // Create calendar resource

            szWorkspacesService.createCalendarResource(szWorkspaceID, rDef);
            System.out.println("Added calendar resource");
        } catch (CwWSEException e) {
            System.out.println("CwWSEException caught while adding " +
                "Calendar resource: "
                + e.getErrorMessage());
        } catch (RemoteException re) {
            System.out.println("RemoteException caught while " +
                "adding Calendar resource: "
                + re.toString());
        }
    }

    try {

        // Create discussion resource

        System.out.println("Adding discussions resource");
        DiscussionResourceDefinition dDef = new DiscussionResourceDefinition();
        dDef.setDescription("Discussion resource");
        dDef.setName("Discussion1");
        szWorkspacesService.createDiscussionResource(szWorkspaceID, dDef);
        System.out.println("Added discussion resource");

    } catch (CwWSEException e2) {
        System.out.println("CwWSEException caught while adding "
            + "Discussion resource: "
            + e2.getErrorMessage());
    } catch (RemoteException re2) {
        System.out.println("RemoteException caught while adding " +
            "Discussion resource: "
            + re2.toString());
    }
}

    try {

        // Create library resource

        System.out.println("Adding library resource");
        LibraryResourceDefinition libDef = new LibraryResourceDefinition();
```

```

libDef.setDescription("Library resource");
libDef.setName("Library1");
libDef.setMappedFolderName("FolderLibrary1");
szWorkspacesService.createLibraryResource(szWorkspaceID, libDef);
System.out.println("Added library resource");

} catch (CwWSEException e3)
{
System.out.println("CwWSEException caught while adding " +
"Library resource: "
+ e3.getErrorMessage());
} catch (RemoteException re3) {
System.out.println("RemoteException caught while adding " +
"Library resource: "
+ re3.toString());
}

try
{

// Create inbox resource

System.out.println("Adding inbox resource");
InboxResourceDefinition iDef = new InboxResourceDefinition();

iDef.setName("My Inbox");
iDef.setEmailInboundPolicy("NONE");
iDef.setEmailSubscriptionFormat(
"My message posted to <a href='${inbox-url}'>Inbox</a> " +
"in workspace <a href='${workspace-url}'>" +
"${workspace-display-name}</a></p><BR>${msg-body}" +
"<BR><a href='${workspaces-home-url}'>");
iDef.setEmailSubscriptionPolicy("disabled");
szWorkspacesService.createEmailResource(szWorkspaceID, iDef);
System.out.println("Added inbox resource");
} catch (CwWSEException e4)
{
System.out.println("CwWSEException caught while adding " +
"inbox resource: "
+ e4.getErrorMessage());
} catch (RemoteException re4) {
System.out.println("RemoteException caught while adding " +
"inbox resource: "
+ re4.toString());
}
}

public static Member[] addMembersToWorkspace(
WorkspacesService szWorkspacesService,
String szWorkspaceID,
User[] szUsersToBeAdded,
String[] szRoles) throws CwWSEException, RemoteException
{
int numberOfUsers = szUsersToBeAdded.length;
Member[] myMembers = new Member[numberOfUsers];

// Create a member from each user
for (int i = 0; i < numberOfUsers; i++)
{

```

```
Member currentMember = new Member();
currentMember.setMemberRoleType(szRoles);
currentMember.setUser(szUsersToBeAdded[i]);
myMembers[i] = currentMember;
}

// Add the members to the workspace
szWorkspacesService.addMembers(szWorkspaceID, myMembers);
System.out.println("Added members");

return myMembers;
}

public static void main(String[] args)
{
    try {

        // Get authentication cookie
        //
        // Ensure that the user has been provisioned for Oracle
        // Mail; Oracle Discussions requires this.

        AuthenticationService myAuthenticationService =
            AuthenticationSample.configureAuthenticationService(
                "rg4",
                "welcome1");

        String authCookie = AuthenticationSample.getAuthenticationCookie
            (myAuthenticationService);

        System.out.println("Retrieved authentication cookie : " + authCookie);

        // Get WorkspacesService and set authentication cookie
        WorkspacesService myWorkspacesService =
            WorkspacesServiceSample.configureWorkspacesService(authCookie);
        // Get UsersService and set authentication cookie
        UsersService myUsersService =
            UsersServiceSample.configureUsersService(authCookie);
        // Retrieve HomeService and set authentication cookie
        HomeService myHomeService =
            HomeServiceSample.configureHomeService(authCookie);

        // Create new workspace (or retrieve it if it already exists)
        WorkspaceItem myWorkspace =
            HomeServiceSample.createOrRetrieveWorkspace(
                myHomeService,
                "NewWorkspace",
                "My New Workspace",
                "Description of My New Workspace",
                "WRITER",
                "ENABLED");

        System.out.println("Retrieved or created workspace");

        // Retrieve Oracle Collaboration Suite users
        // from Oracle Internet Directory from
        // the specified group DN

        User[] myUsers = UsersServiceSample.getUsersFromGroupDN(
            myUsersService,
```

```
"cn=raymond_group,cn=Groups,dc=us,dc=oracle,dc=com");

System.out.println("Retrieved users");

// Make the users members of the workspace and give each
// of them writer role

WorkspacesServiceSample.addMembersToWorkspace(
    myWorkspacesService,
    myWorkspace.getWorkspaceUid(),
    myUsers,
    new String[]{"WRITER"});

System.out.println("Added members to the workspace");

// Create resources

WorkspacesServiceSample.createResources(
    myWorkspacesService,
    myWorkspace.getWorkspaceUid());

System.out.println("Added resources");

myAuthenticationService.logout();

} catch (CwWSEException cwse)
{
    System.out.println("CwWSEException caught: " + cwse.getMessage());
} catch (Exception e)
{
    System.out.println("Exception caught: " + e.toString());
}
}
```



---

---

## Library Service

The Library service allows you to create folders, list quotas, and manage categories, workflow, trash folders, and versioning. The Library service does not allow you to create or upload files.

### Example: Create Folders in and Retrieve Contents of Library

The following sample creates two folders in the library component of a specified workspace, then outputs information about the contents of the library component:

**Example 8-1** *LibraryServiceSample.java*

```
package oracle.sample.workspaces.ws;

import java.rmi.RemoteException;
import javax.xml.rpc.ServiceException;
import oracle.workspaces.ws.AuthenticationService;
import oracle.workspaces.ws.HomeService;
import oracle.workspaces.ws.LibraryService;
import oracle.workspaces.ws.LibraryServiceServiceLocator;
import oracle.workspaces.ws.LibraryServiceSoapBindingStub;
import oracle.workspaces.ws.beans.FilesResourceItem;
import oracle.workspaces.ws.beans.WorkspaceItem;
import oracle.workspaces.ws.exceptions.CwWSEException;
import org.apache.axis.transport.http.HTTPConstants;

public class LibraryServiceSample
{
    public static LibraryService configureLibraryService(
        String szAuthCookie) throws ServiceException
    {
        // Invoke an instance of library service locator and
        // get a handle to library service.
        LibraryServiceServiceLocator lssl = new LibraryServiceServiceLocator();
        LibraryService libService = lssl.getLibraryService();
        ((LibraryServiceSoapBindingStub)libService).
            setMaintainSession(true);
        ((javax.xml.rpc.Stub)libService).
            _setProperty(HTTPConstants.HEADER_COOKIE, szAuthCookie);

        return libService;
    }

    public static FilesResourceItem createFolderInWorkspace(
        LibraryService szLibraryService,
```

```

String szWorkspaceID,
String szFolderName,
String szFolderDesc) throws RemoteException
{
    try {
        return szLibraryService.createFolder(
            szWorkspaceID,
            null, // This folder does not have a parent folder
            szFolderName,
            szFolderDesc
        );
    } catch (CwWSEException e)
    {
        System.out.println("CwWSEException caught while " +
            "creating folder: " + e.getMessage());
    }
    return null;
}

public static FilesResourceItem[] getContentsOfWorkspace(
    LibraryService szLibraryService,
    String szWorkspaceID,
    String szSearchString) throws RemoteException, CwWSEException
{
    // Retrieve the contents of the library component
    // of a specified workspace,
    // then output information about each item found

    FilesResourceItem[] libraryContents =
        szLibraryService.listContents(
            szWorkspaceID, null, szSearchString);

    if (libraryContents == null)
    {
        System.out.println("Library contents are null");
    }
    else {
        for (int i = 0; i < libraryContents.length; i++ )
        {
            System.out.println("Item #" + i + ": " +
                libraryContents[i].getName() + ", " +
                libraryContents[i].getDescription() + ", " +
                libraryContents[i].getId() + ", " +
                libraryContents[i].getResourceItemType());
        }
    }

    return libraryContents;
}

public static void main(String[] args)
{
    try {
        // Get authentication cookie

        AuthenticationService myAuthenticationService =
            AuthenticationSample.configureAuthenticationService(
                "orcladmin",

```

```
        "welcome1");

String authCookie = AuthenticationSample.getAuthenticationCookie
    (myAuthenticationService);

System.out.println("Retrieved authentication cookie : " + authCookie);

// Get LibraryService and set authentication cookie

LibraryService myLibraryService =
    LibraryServiceSample.configureLibraryService(authCookie);

// Retrieve HomeService and set authentication cookie
HomeService myHomeService =
    HomeServiceSample.configureHomeService(authCookie);

// Create new workspace (or retrieve it if it already exists)
WorkspaceItem myWorkspace =
    HomeServiceSample.createOrRetrieveWorkspace(
        myHomeService,
        "NewWorkspace",
        "My New Workspace",
        "Description of My New Workspace",
        "WRITER",
        "ENABLED");

System.out.println("Retrieved or created workspace");

// Create some folders in the library resource

LibraryServiceSample.createFolderInWorkspace(
    myLibraryService,
    myWorkspace.getWorkspaceUid(),
    "New Folder 1",
    "Description of New Folder 1");

System.out.println("Created 'New Folder 1'");

LibraryServiceSample.createFolderInWorkspace(
    myLibraryService,
    myWorkspace.getWorkspaceUid(),
    "New Folder 2",
    "Description of New Folder 2");

System.out.println("Created 'New Folder 2'");

// Output contents of library resource

System.out.println("Contents of library component " +
    "of workspace:");

LibraryServiceSample.getContentsOfWorkspace(
    myLibraryService,
    myWorkspace.getWorkspaceUid(),
    "*");

myAuthenticationService.logout();

} catch (CwWSException cwwse)
{
```

```
        System.out.println("CwWSEException caught: " + cwwse.getMessage());
    } catch (Exception e)
    {
        System.out.println("Exception caught: " + e.toString());
    }
}
}
```

---

---

## Meetings Service

The Meetings service allows you to create, update, retrieve, list, and delete meetings and get the free/busy status of attendees of a particular meeting.

### Example: Create Meeting

The following sample creates a meeting of duration one hour scheduled for the current time:

#### **Example 9-1** *MeetingsServiceSample.java*

```
package oracle.sample.workspaces.ws;

import java.rmi.RemoteException;
import java.util.Calendar;
import java.util.GregorianCalendar;
import javax.xml.rpc.ServiceException;
import oracle.workspaces.ws.AuthenticationService;
import oracle.workspaces.ws.HomeService;
import oracle.workspaces.ws.MeetingsService;
import oracle.workspaces.ws.MeetingsServiceServiceLocator;
import oracle.workspaces.ws.MeetingsServiceSoapBindingStub;
import oracle.workspaces.ws.beans.MeetingDefinition;
import oracle.workspaces.ws.beans.MeetingResourceItem;
import oracle.workspaces.ws.beans.WorkspaceItem;
import oracle.workspaces.ws.exceptions.CwWSEException;
import org.apache.axis.transport.http.HTTPConstants;

public class MeetingsServiceSample
{

    public static MeetingsService configureMeetingsService(
        String szAuthCookie) throws ServiceException
    {
        MeetingsServiceServiceLocator meetingSSL =
            new MeetingsServiceServiceLocator();
        MeetingsService meetService = meetingSSL.getMeetingsService();
        ((MeetingsServiceSoapBindingStub)meetService).
            setMaintainSession(true);
        ((javax.xml.rpc.Stub)meetService).
            _setProperty(HTTPConstants.HEADER_COOKIE, szAuthCookie);
        return meetService;
    }

    public static MeetingResourceItem createMeetingForToday(
        MeetingsService szMeetingsService,
```

```

    String szWorkspaceID) throws RemoteException, CwWSEException
    {
        // Create meeting of duration one hour scheduled
        // for the current time

        MeetingDefinition mtngDef = new MeetingDefinition();
        mtngDef.setName("My Meeting1");
        mtngDef.setLocation("meetingRoom1");
        mtngDef.setDescription("Meeting1 description");
        Calendar today = new GregorianCalendar();
        Calendar todayAnHourLater = new GregorianCalendar();
        todayAnHourLater.add(Calendar.HOUR_OF_DAY, 1);
        mtngDef.setStartTime(today);
        mtngDef.setEndTime(todayAnHourLater);
        mtngDef.setEventType("APPOINTMENT");
        mtngDef.setAttendingType("ALL");

        // Invoke method to create a meeting
        return szMeetingsService.createMeeting(szWorkspaceID, mtngDef);
    }

public static void main(String[] args)
{
    try {
        // Get authentication cookie
        AuthenticationService myAuthenticationService =
            AuthenticationSample.configureAuthenticationService(
                "rg4",
                "welcome1");

        String authCookie = AuthenticationSample.getAuthenticationCookie
            (myAuthenticationService);

        System.out.println("Retrieved authentication cookie : " + authCookie);

        // Get MeetingsService and set authentication cookie
        MeetingsService myMeetingsService =
            MeetingsServiceSample.configureMeetingsService(authCookie);

        // Retrieve HomeService and set authentication cookie
        HomeService myHomeService =
            HomeServiceSample.configureHomeService(authCookie);

        // Create new workspace (or retrieve it if it already exists)
        WorkspaceItem myWorkspace =
            HomeServiceSample.createOrRetrieveWorkspace(
                myHomeService,
                "NewWorkspace",
                "My New Workspace",
                "Description of My New Workspace",
                "WRITER",
                "ENABLED");

        System.out.println("Retrieved or created workspace");

        // Create a meeting

        MeetingsServiceSample.createMeetingForToday(
            myMeetingsService,
            myWorkspace.getWorkspaceUid());
    }
}

```

```
        System.out.println("Created new meeting");

        myAuthenticationService.logout();

    } catch (CwWSEException cwse)
    {
        System.out.println("CwWSEException caught: " + cwse.toString());
        cwse.printStackTrace();
    } catch (Exception e)
    {
        System.out.println("Exception caught: " + e.toString());
    }
}
}
```



The Tasks service allows you create, update, retrieve, list, and delete tasks and show which tasks are expired or unexpired.

### Example: Create and Search for Tasks

The following sample creates three tasks of different start times (the first and second task have start times earlier than the third). The sample then searches for tasks that have start times within the first task's start time and the second task's start time. As a result, the third task will not appear in the search results.

#### *Example 10-1 TasksServiceSample.java*

```
package oracle.sample.workspaces.ws;

import java.rmi.RemoteException;
import java.util.Calendar;
import javax.xml.rpc.ServiceException;
import oracle.workspaces.ws.AuthenticationService;
import oracle.workspaces.ws.HomeService;
import oracle.workspaces.ws.TasksService;
import oracle.workspaces.ws.TasksServiceServiceLocator;
import oracle.workspaces.ws.TasksServiceSoapBindingStub;
import oracle.workspaces.ws.beans.TaskAssignee;
import oracle.workspaces.ws.beans.TaskDefinition;
import oracle.workspaces.ws.beans.TaskResourceItem;
import oracle.workspaces.ws.beans.WorkspaceItem;
import oracle.workspaces.ws.exceptions.CwWSEException;
import org.apache.axis.transport.http.HTTPConstants;

public class TasksServiceSample
{
    public static TasksService configureTasksService(
        String szAuthCookie) throws ServiceException
    {
        TasksServiceServiceLocator tssl =
            new TasksServiceServiceLocator();
        TasksService tService = tssl.getTasksService();
        ((TasksServiceSoapBindingStub)tService).setMaintainSession(true);
        ((javax.xml.rpc.Stub)tService).
            _setProperty(HTTPConstants.HEADER_COOKIE, szAuthCookie);
        return tService;
    }

    public static TaskResourceItem createTask(
```

```
TasksService szTasksService,
String szWorkspaceID,
String szTaskName,
String szTaskDesc,
String szPriority,
Calendar szStartTime,
Calendar szEndTime,
int szPercentComplete,
String szUserNickname)
throws RemoteException, CwWSEException
{
    TaskDefinition td1 = new TaskDefinition();
    td1.setName(szTaskName);
    td1.setDescription(szTaskDesc);
    td1.setStartTime(szStartTime);
    td1.setEndTime(szEndTime);
    td1.setPriority(szPriority);
    td1.setPercentComplete(szPercentComplete);
    TaskAssignee tal = new TaskAssignee();
    tal.setAssignee(szUserNickname);
    td1.setAssigneeList(new TaskAssignee[]{tal});

    return szTasksService.createTask(szWorkspaceID, td1);
}

public static TaskResourceItem[] outputTasksByDateRange(
    TasksService szTasksService,
    String szWorkspaceID,
    Calendar startDate,
    Calendar endDate) throws RemoteException, CwWSEException
{
    TaskResourceItem[] currentTasks = szTasksService.listTasksByRange(
        szWorkspaceID,
        startDate,
        endDate,
        true);

    if (currentTasks == null)
    {
        System.out.println("No tasks found");
    } else {
        for (int i=0; i<currentTasks.length; i++)
        {
            System.out.println("Task #" + i +
                ", Name: "+ currentTasks[i].getName());
        }
    }
    return currentTasks;
}

public static void main(String[] args)
{
    String myUsername = "rg4";
    String myPassword = "welcome1";
    try {
        // Get authentication cookie

        AuthenticationService myAuthenticationService =
            AuthenticationSample.configureAuthenticationService(
                myUsername,
```

```
        myPassword);

String authCookie = AuthenticationSample.getAuthenticationCookie
    (myAuthenticationService);

System.out.println("Retrieved authentication cookie : " + authCookie);

// Get TasksService and set authentication cookie
TasksService myTasksService =
    TasksServiceSample.configureTasksService(authCookie);

// Retrieve HomeService and set authentication cookie
HomeService myHomeService =
    HomeServiceSample.configureHomeService(authCookie);

// Create new workspace (or retrieve it if it already exists)
WorkspaceItem myWorkspace =
    HomeServiceSample.createOrRetrieveWorkspace(
        myHomeService,
        "NewWorkspace",
        "My New Workspace",
        "Description of My New Workspace",
        "WRITER",
        "ENABLED");

System.out.println("Retrieved or created workspace");

// Create some tasks

Calendar startTime1 = Calendar.getInstance();
Calendar endTime1 = Calendar.getInstance();
Calendar startTime2 = Calendar.getInstance();
Calendar endTime2 = Calendar.getInstance();
Calendar startTime3 = Calendar.getInstance();
Calendar endTime3 = Calendar.getInstance();

// Ignore seconds

startTime1.set(Calendar.SECOND, 0);
endTime1.set(Calendar.SECOND, 0);
startTime2.set(Calendar.SECOND, 0);
endTime2.set(Calendar.SECOND, 0);
startTime3.set(Calendar.SECOND, 0);
endTime3.set(Calendar.SECOND, 0);

endTime1.add(Calendar.HOUR_OF_DAY, 6);
endTime2.add(Calendar.HOUR_OF_DAY, 6);
endTime3.add(Calendar.HOUR_OF_DAY, 6);

startTime2.add(Calendar.DAY_OF_YEAR, 7);
endTime2.add(Calendar.DAY_OF_YEAR, 7);
startTime3.add(Calendar.DAY_OF_YEAR, 14);
endTime3.add(Calendar.DAY_OF_YEAR, 14);

TasksServiceSample.createTask(
    myTasksService,
    myWorkspace.getWorkspaceUid(),
    "My first task",
    "My first task description",
    "PRIORITY_NORMAL",
```

```
        startTime1,
        endTime1,
        50,
        myUsername);

System.out.println("Created first task");

TasksServiceSample.createTask(
    myTasksService,
    myWorkspace.getWorkspaceUid(),
    "My second task",
    "My second task description",
    "PRIORITY_HIGH",
    startTime2,
    endTime2,
    25,
    myUsername);

System.out.println("Created second task");

TasksServiceSample.createTask(
    myTasksService,
    myWorkspace.getWorkspaceUid(),
    "My third task",
    "My third task description",
    "PRIORITY_LOW",
    startTime3,
    endTime3,
    10,
    myUsername);

System.out.println("Created third task");

// Retrieve tasks between the date range of
// startTime1 and startTime2.
// As a result, the third task will not
// be retrieved.

TasksServiceSample.outputTasksByDateRange(
    myTasksService,
    myWorkspace.getWorkspaceUid(),
    startTime1,
    startTime2);

myAuthenticationService.logout();

} catch (CwWSEException cwse)
{
    System.out.println("CwWSEException caught: " + cwse.toString());
    cwse.printStackTrace();
} catch (Exception e)
{
    System.out.println("Exception caught: " + e.toString());
}
}
```

---

---

## Announcements Service

The Announcements service allows users to create, update, retrieve, and delete announcements. An announcement is a message posted by a member of a workspace that all members of that workspace may see on the workspace home page.

### Example: Create Announcement

The following sample creates an announcement that will expire seven days from the current date:

**Example 11-1** *AnnouncementsServiceSample.java*

```
package oracle.sample.workspaces.ws;
import java.rmi.RemoteException;
import java.util.Calendar;
import javax.xml.rpc.ServiceException;
import oracle.workspaces.ws.AnnouncementsService;
import oracle.workspaces.ws.AnnouncementsServiceServiceLocator;
import oracle.workspaces.ws.AnnouncementsServiceSoapBindingStub;
import oracle.workspaces.ws.AuthenticationService;
import oracle.workspaces.ws.HomeService;
import oracle.workspaces.ws.beans.AnnouncementDefinition;
import oracle.workspaces.ws.beans.AnnouncementItem;
import oracle.workspaces.ws.beans.WorkspaceItem;
import oracle.workspaces.ws.exceptions.CwWSEException;
import org.apache.axis.transport.http.HTTPConstants;

public class AnnouncementsServiceSample
{
    public static AnnouncementsService configureAnnouncementsService(
        String szAuthCookie) throws ServiceException
    {
        AnnouncementsServiceServiceLocator assl =
            new AnnouncementsServiceServiceLocator();
        AnnouncementsService aService = assl.getAnnouncementsService();
        ((AnnouncementsServiceSoapBindingStub)aService).
            setMaintainSession(true);
        ((javax.xml.rpc.Stub)aService).
            _setProperty(HTTPConstants.HEADER_COOKIE, szAuthCookie);
        return aService;
    }

    public static AnnouncementItem createAnnouncement(
        AnnouncementsService szAnnouncementsService,
        String szWorkspaceID,
        String szTitle,
```

```
        String szBody) throws RemoteException, CwWSEException
    {
Calendar expirationTime = Calendar.getInstance();
expirationTime.set(Calendar.SECOND, 0);
expirationTime.add(Calendar.DAY_OF_YEAR, 7);
AnnouncementDefinition annDefn1 = new AnnouncementDefinition();
annDefn1.setTitle(szTitle);
annDefn1.setBody(szBody);
annDefn1.setExpirationTime(expirationTime);
annDefn1.setPlainText(true);

return szAnnouncementsService.createAnnouncement(
    szWorkspaceID,
    annDefn1);
}

public static void main(String[] args)
{
    try {

        // Get authentication cookie

        AuthenticationService myAuthenticationService =
            AuthenticationSample.configureAuthenticationService(
                "rg4",
                "welcome1");

        String authCookie = AuthenticationSample.getAuthenticationCookie
            (myAuthenticationService);

        System.out.println("Retrieved authentication cookie : " + authCookie);

        // Retrieve AnnouncementsService and set authentication cookie
        AnnouncementsService myAnnouncementsService =
            AnnouncementsServiceSample.
                configureAnnouncementsService(authCookie);

        // Retrieve HomeService and set authentication cookie
        HomeService myHomeService =
            HomeServiceSample.configureHomeService(authCookie);

        // Create new workspace (or retrieve it if it already exists)
        WorkspaceItem myWorkspace =
            HomeServiceSample.createOrRetrieveWorkspace(
                myHomeService,
                "NewWorkspace",
                "My New Workspace",
                "Description of My New Workspace",
                "WRITER",
                "ENABLED");

        System.out.println("Retrieved or created workspace");

        // Create an announcement that will expire seven
// days from the current date

        AnnouncementsServiceSample.createAnnouncement(
            myAnnouncementsService,
            myWorkspace.getWorkspaceUid(),
            "My announcement title",
```

```
        "My announcement body");

        System.out.println("Created announcement");

        myAuthenticationService.logout();

    } catch (CwWSEException cwse)
    {
        System.out.println("CwWSEException caught: " + cwse.getMessage());
    } catch (Exception e)
    {
        System.out.println("Exception caught: " + e.toString());
    }
}
}
```



---

---

## Discussion Service

The Discussion service provides methods for working with discussions resources.

Three kinds of operations are available:

- Board operations allow you to create, update, delete, and retrieve boards (which are called forums in the Oracle Workspaces user interface).
- Thread operations allow you to retrieve and delete messages within a thread, add and remove messages within a thread to or from a user's favorites, and retrieve popular threads. A thread is an aggregation of related messages. A thread is created when a new message is posted on a board, and that message is not a reply to an existing message.
- Message operations allow you to post, retrieve, and delete messages.

### Example: Create Board, Post Message, and Post Reply

The following sample creates a board in a specified workspace, or retrieves the board if one with the same name already exists. It then posts a message on the board, then posts a reply to that message:

#### *Example 12-1 DiscussionServiceSample.java*

```
package oracle.sample.workspaces.ws;

import java.rmi.RemoteException;
import javax.xml.rpc.ServiceException;
import oracle.workspaces.ws.AuthenticationService;
import oracle.workspaces.ws.DiscussionService;
import oracle.workspaces.ws.DiscussionServiceServiceLocator;
import oracle.workspaces.ws.DiscussionServiceSoapBindingStub;
import oracle.workspaces.ws.HomeService;
import oracle.workspaces.ws.beans.BoardDefinition;
import oracle.workspaces.ws.beans.BoardItem;
import oracle.workspaces.ws.beans.MessageDefinition;
import oracle.workspaces.ws.beans.MessageItem;
import oracle.workspaces.ws.beans.ThreadItem;
import oracle.workspaces.ws.beans.WorkspaceItem;
import oracle.workspaces.ws.exceptions.CwWSEException;
import org.apache.axis.transport.http.HTTPConstants;

public class DiscussionServiceSample
{
    public static DiscussionService configureDiscussionService(
        String szAuthCookie) throws ServiceException
    {
```

```

DiscussionServiceServiceLocator dssl =
    new DiscussionServiceServiceLocator();
DiscussionService discussionService =
    dssl.getDiscussionService();
((DiscussionServiceSoapBindingStub)discussionService).
    setMaintainSession(true);
((javax.xml.rpc.Stub) discussionService).
    _setProperty(HTTPConstants.HEADER_COOKIE, szAuthCookie);
return discussionService;
}

public static MessageItem createMessage(
    DiscussionService szDiscussionService,
    String szBoardID,
    String szWorkspaceID) throws RemoteException, CwWSEException
{
    MessageItem myMessage = null;
    String[] mAttrib = {"CONTENT", "CONTENT_TYPE", "SIZE",
        "LEVEL_IN_THREAD", "IS_MARKED_DELETED",
        "PRIORITY", "HAS_ATTACHMENTS"};

    MessageDefinition msgDef = new MessageDefinition();
    msgDef.setContentType("text/plain");
    msgDef.setText("Text of my new message2");
    msgDef.setSubject("Subject of my new message2");
    myMessage = szDiscussionService.createMessage(
        szWorkspaceID,
        szBoardID,
        null, // Create a new thread with this message
        msgDef,
        mAttrib);

    return myMessage;
}

public static MessageItem postReply(
    DiscussionService szDiscussionService,
    String szBoardID,
    String szWorkspaceID,
    String szMessageID,
    String szThreadID) throws RemoteException, CwWSEException
{
    // Post a reply to the message specified by szMessageID

    MessageItem myMessage = null;
    String[] mAttrib = {"CONTENT", "CONTENT_TYPE", "SIZE",
        "LEVEL_IN_THREAD", "IS_MARKED_DELETED",
        "PRIORITY", "HAS_ATTACHMENTS"};

    MessageDefinition msgDef = new MessageDefinition();
    msgDef.setContentType("text/plain");
    msgDef.setText("Text of my reply2");
    msgDef.setSubject("Subject of my reply2");
    msgDef.setMessageRepliedTo(szMessageID);

    myMessage = szDiscussionService.createMessage(
        szWorkspaceID,
        szBoardID,
        szThreadID,
        msgDef,

```

```

        mAttrib);

    return myMessage;
}

public static BoardItem createDiscussionBoard(
    DiscussionService szDiscussionService,
    String szWorkspaceID) throws RemoteException, CwWSEException
{
    BoardItem myBoard = null;

    // Create a board definition object and set its attributes.
    BoardDefinition boardDef = new BoardDefinition();
    boardDef.setDescription("Board1 desc");
    boardDef.setName("Board1 name");
    boardDef.setDisplayName("Board1 name");
    boardDef.setBoardEditDeletePolicy("NONE");
    boardDef.setEmailInboundPolicy("NONE");
    String[] attrib = {"EDIT_DELETE_POLICY", "INBOUND_POLICY"};

    // Invoke method to create board within workspace.

    try {
        myBoard = szDiscussionService.createBoard(
            szWorkspaceID,
            boardDef,
            attrib);
    } catch (CwWSEException cwse)
    {
        // createBoard will throw an exception if a board
        // of the same name already exists.
        //
        // Attempt to retrieve the board with the same name

        myBoard = szDiscussionService.getBoardByName(
            szWorkspaceID,
            "Board1 name",
            null);
    }
    return myBoard;
}

public static void main(String[] args)
{
    try {
        // Get authentication cookie

        AuthenticationService myAuthenticationService =
            AuthenticationSample.configureAuthenticationService(
                "rg4",
                "welcome1");

        String authCookie = AuthenticationSample.getAuthenticationCookie(
            myAuthenticationService);

        System.out.println("Retrieved authentication cookie : " + authCookie);

        // Get DiscussionService and set authentication cookie
    }
}

```

```

DiscussionService myDiscussionService =
    DiscussionServiceSample.configureDiscussionService(authCookie);

// Retrieve HomeService and set authentication cookie
HomeService myHomeService =
    HomeServiceSample.configureHomeService(authCookie);

// Create new workspace (or retrieve it if it already exists)
WorkspaceItem myWorkspace =
    HomeServiceSample.createOrRetrieveWorkspace(
        myHomeService,
        "NewWorkspace",
        "My New Workspace",
        "Description of My New Workspace",
        "WRITER",
        "ENABLED");

System.out.println("Retrieved or created workspace");

BoardItem myBoard = DiscussionServiceSample.createDiscussionBoard(
    myDiscussionService,
    myWorkspace.getWorkspaceUid());

System.out.println("Created or retrieved discussion board");

// Create a new message

MessageItem myMessage = DiscussionServiceSample.createMessage(
    myDiscussionService,
    myBoard.getId(),
    myWorkspace.getWorkspaceUid());

System.out.println("Created message");

// Post a reply to that message

DiscussionServiceSample.postReply(
    myDiscussionService,
    myBoard.getId(),
    myWorkspace.getWorkspaceUid(),
    myMessage.getId(),
    myMessage.getThreadUid());

System.out.println("Posted a reply");

myAuthenticationService.logout();

} catch (CwWSEException cwwse)
{
    System.out.println("CwWSEException caught: " + cwwse.getMessage());
    cwwse.printStackTrace();
} catch (Exception e)
{
    System.out.println("Exception caught: " + e.toString());
}
}
}

```

The Inbox service allows you to create, retrieve, and delete messages or threads within an inbox resource.

### Example: Post Message to Inbox

The following sample posts a message to the workspace's inbox:

**Example 13-1** *InboxServiceSample.java*

```
package oracle.sample.workspaces.ws;

import java.rmi.RemoteException;
import javax.xml.rpc.ServiceException;
import oracle.workspaces.ws.AuthenticationService;
import oracle.workspaces.ws.HomeService;
import oracle.workspaces.ws.InboxService;
import oracle.workspaces.ws.InboxServiceServiceLocator;
import oracle.workspaces.ws.InboxServiceSoapBindingStub;
import oracle.workspaces.ws.beans.MessageDefinition;
import oracle.workspaces.ws.beans.MessageItem;
import oracle.workspaces.ws.beans.WorkspaceItem;
import oracle.workspaces.ws.exceptions.CwWSEException;
import org.apache.axis.transport.http.HTTPConstants;

public class InboxServiceSample
{
    public static InboxService configureInboxService(
        String szAuthCookie) throws ServiceException
    {
        InboxServiceServiceLocator issl =
            new InboxServiceServiceLocator();
        InboxService iService =
            issl.getInboxService();
        ((InboxServiceSoapBindingStub)iService).
            setMaintainSession(true);
        ((javax.xml.rpc.Stub)iService).
            _setProperty(HTTPConstants.HEADER_COOKIE, szAuthCookie);

        return iService;
    }

    public static MessageItem postMessage(
        InboxService szInboxService,
        String szWorkspaceID
    ) throws RemoteException, CwWSEException
```

```
{
MessageDefinition msgDefn1 = new MessageDefinition();
msgDefn1.setContentType("text/plain");
msgDefn1.setSubject("Mail message subject");
msgDefn1.setText("This is the first message to this email inbox");

return szInboxService.createMessage(
    szWorkspaceID,
    null, // No thread specified
    msgDefn1,
    null); // No attributes are to be retrieved from message
}

public static void main(String[] args)
{
    try {
        // Get authentication cookie

        AuthenticationService myAuthenticationService =
            AuthenticationSample.configureAuthenticationService(
                "rg4",
                "welcome1");

        String authCookie = AuthenticationSample.getAuthenticationCookie
            (myAuthenticationService);

        System.out.println("Retrieved authentication cookie : " + authCookie);

        // Get InboxService and set authentication cookie
        InboxService myInboxService =
            InboxServiceSample.configureInboxService(authCookie);

        // Retrieve HomeService and set authentication cookie
        HomeService myHomeService =
            HomeServiceSample.configureHomeService(authCookie);

        // Create new workspace (or retrieve it if it already exists)
        WorkspaceItem myWorkspace =
            HomeServiceSample.createOrRetrieveWorkspace(
                myHomeService,
                "NewWorkspace",
                "My New Workspace",
                "Description of My New Workspace",
                "WRITER",
                "ENABLED");

        System.out.println("Retrieved or created workspace");

        // Create a message in the inbox resource of the
        // workspace

        InboxServiceSample.postMessage(
            myInboxService,
            myWorkspace.getWorkspaceUid());

        System.out.println("Posted message");

        myAuthenticationService.logout();

    } catch (CwWSEException cwwse)
```

```
{
    System.out.println("CwWSEException caught: " + cwse.toString());
    cwse.printStackTrace();
} catch (Exception e)
{
    System.out.println("Exception caught: " + e.toString());
}
}
```



---

---

## Views Service

The Views service allows you to create, update, retrieve, and delete views. Use views to group together and conveniently view disparate workspace content on one page.

### Example: Create View, Add Items to View, and Search for Views that Contain Item

The following sample creates a meeting and a board in a workspace, creates a view, adds the meeting and board to the view, and outputs the views that contain the board:

#### *Example 14-1 ViewsServiceSample.java*

```
package oracle.sample.workspaces.ws;

import java.rmi.RemoteException;
import javax.xml.rpc.ServiceException;
import oracle.workspaces.ws.AuthenticationService;
import oracle.workspaces.ws.DiscussionService;
import oracle.workspaces.ws.HomeService;
import oracle.workspaces.ws.MeetingsService;
import oracle.workspaces.ws.ViewsService;
import oracle.workspaces.ws.ViewsServiceServiceLocator;
import oracle.workspaces.ws.ViewsServiceSoapBindingStub;
import oracle.workspaces.ws.beans.BoardItem;
import oracle.workspaces.ws.beans.MeetingResourceItem;
import oracle.workspaces.ws.beans.ViewItemDefinition;
import oracle.workspaces.ws.beans.ViewResourceItem;
import oracle.workspaces.ws.beans.WorkspaceItem;
import oracle.workspaces.ws.exceptions.CwWSEException;
import org.apache.axis.transport.http.HTTPConstants;

public class ViewsServiceSample
{
    public static ViewsService configureViewsService(
        String szAuthCookie) throws ServiceException
    {
        ViewsServiceServiceLocator vssl =
            new ViewsServiceServiceLocator();
        ViewsService vService =
            vssl.getViewsService();
        ((ViewsServiceSoapBindingStub)vService).
            setMaintainSession(true);
        ((javax.xml.rpc.Stub)vService).
            _setProperty(HTTPConstants.HEADER_COOKIE, szAuthCookie);
    }
}
```

```

        return vService;
    }

    public static ViewResourceItem createView(
        ViewsService szViewsService,
        String szWorkspaceID,
        String szName,
        String szDesc,
        String[] szItems) throws RemoteException, CwWSEException
    {
        ViewResourceItem currentView = null;
        ViewItemDefinition viewDef = new ViewItemDefinition();
        viewDef.setName(szName);
        viewDef.setDescription(szDesc);

        // Create view

        try {

            currentView = szViewsService.createView(
                szWorkspaceID,
                viewDef);
        } catch (CwWSEException e)
        {
            // Retrieve view if it already exists

            ViewResourceItem[] allViews =
                szViewsService.listViews(szWorkspaceID);

            if (allViews != null)
            {
                for (int i=0; i<allViews.length; i++)
                {
                    if (allViews[i].getName().equalsIgnoreCase(szName))
                    {
                        currentView = szViewsService.getViewById(
                            szWorkspaceID,
                            allViews[i].getId());
                        break;
                    }
                }
            }
        }

        System.out.println("Created view");

        // Add items to view

        szViewsService.addItem(
            szWorkspaceID,
            currentView.getId(),
            szItems);

        System.out.println("Added items to view");

        return currentView;
    }

    public static ViewResourceItem[] outputParentViews(
        ViewsService szViewsService,

```

```

String szWorkspaceID,
String szItemId) throws RemoteException, CwWSException
{
    ViewResourceItem[] currentViews =
        szViewsService.listParentViews(szWorkspaceID, szItemId);
    if (currentViews == null)
    {
        System.out.println("No views found");
    } else
    {
        System.out.println("Views containing specified item:");
        for (int i=0; i<currentViews.length; i++)
        {
            System.out.println("View #" + i + " Name: " +
                currentViews[i].getName());
        }
    }
    return currentViews;
}

public static void main(String[] args)
{
    try {
        // Get authentication cookie

        AuthenticationService myAuthenticationService =
            AuthenticationSample.configureAuthenticationService(
                "rg4",
                "welcome1");

        String authCookie = AuthenticationSample.getAuthenticationCookie
            (myAuthenticationService);

        System.out.println("Retrieved authentication cookie : " + authCookie);

        // Get ViewsService and set authentication cookie
        ViewsService myViewsService =
            ViewsServiceSample.configureViewsService(authCookie);

        // Get MeetingsService and set authentication cookie
        MeetingsService myMeetingsService =
            MeetingsServiceSample.configureMeetingsService(authCookie);

        // Get DiscussionService and set authentication cookie

        DiscussionService myDiscussionService =
            DiscussionServiceSample.configureDiscussionService(authCookie);

        // Retrieve HomeService and set authentication cookie
        HomeService myHomeService =
            HomeServiceSample.configureHomeService(authCookie);

        // Create new workspace (or retrieve it if it already exists)
        WorkspaceItem myWorkspace =
            HomeServiceSample.createOrRetrieveWorkspace(
                myHomeService,
                "NewWorkspace",
                "My New Workspace",
                "Description of My New Workspace",

```

```

        "WRITER",
        "ENABLED");

System.out.println("Retrieved or created workspace");

// Create a board

BoardItem myBoard = DiscussionServiceSample.createDiscussionBoard(
    myDiscussionService,
    myWorkspace.getWorkspaceUid());

System.out.println("Created board");

// Create a meeting

MeetingResourceItem myMeeting =
    MeetingsServiceSample.createMeetingForToday(
        myMeetingsService,
        myWorkspace.getWorkspaceUid());

System.out.println("Created meeting");

// Create a new view, and put the board
// and meeting into that view

String myBoardID = myBoard.getId();
String myMeetingID = myMeeting.getId();

String[] myItems = new String[]{
    myBoardID,
    myMeetingID};

ViewsServiceSample.createView(
    myViewsService,
    myWorkspace.getWorkspaceUid(),
    "My view",
    "My view description",
    myItems);

System.out.println("Created view with items");

// Output the views that contain the board

ViewsServiceSample.outputParentViews(
    myViewsService,
    myWorkspace.getWorkspaceUid(),
    myBoardID);

myAuthenticationService.logout();

} catch (CwWSEException cwWse)
{
    System.out.println("CwWSEException caught: " + cwWse.toString());
    cwWse.printStackTrace();
} catch (Exception e)
{
    System.out.println("Exception caught: " + e.toString());
}
}

```

}



---

---

## Attachments Service

An *attachment* (which is called a link in the Oracle Workspace user interface) is a mechanism that associates resource items with each other. The following are some of the tasks that you may perform with the Attachments service:

- Attach one Oracle Workspaces resource item with another. For example you may attach a discussion thread to a file.
- Attach multiple resource items to a resource item.
- Query the attachments attached to a particular resource item (target attachments).
- Query the resource item to which an entity is attached (source attachments).

### Example: Create Attachment

The following sample creates a board (or retrieves it if it already exists), creates a meeting, and creates an attachment that associates the board to the meeting:

**Example 15–1 AttachmentsServiceSample.java**

```
package oracle.sample.workspaces.ws;

import java.rmi.RemoteException;
import javax.xml.rpc.ServiceException;
import oracle.workspaces.ws.AttachmentsService;
import oracle.workspaces.ws.AttachmentsServiceServiceLocator;
import oracle.workspaces.ws.AttachmentsServiceSoapBindingStub;
import oracle.workspaces.ws.AuthenticationService;
import oracle.workspaces.ws.DiscussionService;
import oracle.workspaces.ws.HomeService;
import oracle.workspaces.ws.MeetingsService;
import oracle.workspaces.ws.beans.AttachmentDefinition;
import oracle.workspaces.ws.beans.BoardItem;
import oracle.workspaces.ws.beans.MeetingResourceItem;
import oracle.workspaces.ws.beans.WorkspaceItem;
import oracle.workspaces.ws.exceptions.CwWSEException;
import org.apache.axis.transport.http.HTTPConstants;

public class AttachmentsServiceSample
{
    public static AttachmentsService configureAttachmentsService(
        String szAuthCookie) throws ServiceException
    {
        AttachmentsServiceServiceLocator assl =
            new AttachmentsServiceServiceLocator();
        AttachmentsService aService =
```

```

        assl.getAttachmentsService();
    ((AttachmentsServiceSoapBindingStub)aService).
        setMaintainSession(true);
    ((javax.xml.rpc.Stub) aService).
        _setProperty(HTTPConstants.HEADER_COOKIE, szAuthCookie);
    return aService;
}

public static void addAttachment(
    AttachmentsService szAttachmentsService,
    String szWorkspaceID,
    String szItemBeingAttachedID,
    String szItemReceivingAttachmentID)
    throws RemoteException, CwWSEException
{
    // szItemBeingAttachedID is the UID of the entity
    // being attached
    //
    // szItemReceivingAttachmentID is the UID of
    // the entity to which the attachment is being added

    AttachmentDefinition[] aDefArray = new AttachmentDefinition[1];
    AttachmentDefinition aDef1 = new AttachmentDefinition();
    aDef1.setDescription("Adding attachments as links");
    aDef1.setEntityId(szItemBeingAttachedID);
    aDefArray[0] = aDef1;
    szAttachmentsService.addAttachments(
        szWorkspaceID,
        szItemReceivingAttachmentID,
        aDefArray);
}

public static void main(String[] args)
{
    try {
        // Get authentication cookie

        AuthenticationService myAuthenticationService =
            AuthenticationSample.configureAuthenticationService(
                "rg4",
                "welcome1");

        String authCookie = AuthenticationSample.getAuthenticationCookie
            (myAuthenticationService);

        System.out.println("Retrieved authentication cookie : " + authCookie);

        // Get AttachmentsService and set authentication cookie
        AttachmentsService myAttachmentsService =
            AttachmentsServiceSample.configureAttachmentsService(authCookie);

        // Get DiscussionService and set authentication cookie
        DiscussionService myDiscussionService =
            DiscussionServiceSample.configureDiscussionService(authCookie);

        // Get MeetingsService and set authentication cookie
        MeetingsService myMeetingsService =
            MeetingsServiceSample.configureMeetingsService(authCookie);
    }
}

```

```
// Retrieve HomeService and set authentication cookie
HomeService myHomeService =
    HomeServiceSample.configureHomeService(authCookie);

// Create new workspace (or retrieve it if it already exists)
WorkspaceItem myWorkspace =
    HomeServiceSample.createOrRetrieveWorkspace(
        myHomeService,
        "NewWorkspace",
        "My New Workspace",
        "Description of My New Workspace",
        "WRITER",
        "ENABLED");

System.out.println("Retrieved or created workspace");

// Create a new meeting

MeetingResourceItem myMeeting =
    MeetingsServiceSample.createMeetingForToday(
        myMeetingsService,
        myWorkspace.getWorkspaceUid());

System.out.println("Created new meeting");

// Create (or retrieve) a board

BoardItem myBoard =
    DiscussionServiceSample.createDiscussionBoard(
        myDiscussionService,
        myWorkspace.getWorkspaceUid());

System.out.println("Created or retrieved board");

// The following method call will create a link from
// a board to a meeting

AttachmentsServiceSample.addAttachment(
    myAttachmentsService,
    myWorkspace.getWorkspaceUid(),
    myMeeting.getId(),
    myBoard.getId());

System.out.println("Created attachment");

myAuthenticationService.logout();

} catch (CwWSEException cwse)
{
    System.out.println("CwWSEException caught: " + cwse.toString());
    cwse.printStackTrace();
} catch (Exception e)
{
    System.out.println("Exception caught: " + e.toString());
}
}
```



---

---

## Template Service

The Template service allows users with application administrator role to create, retrieve, delete, and store workspace templates.

A template is an XML document that is used to create new workspaces with some predefined settings and content. See "Developing Workspace Templates with XML" in *Oracle Workspaces Application Developer's Guide* for more information.

### Example: Retrieve Templates

The following sample outputs information about each workspace template stored in Oracle Workspaces:

#### **Example 16-1** *TemplateServiceSample.java*

```
package oracle.sample.workspaces.ws;

import java.rmi.RemoteException;
import javax.xml.rpc.ServiceException;
import oracle.workspaces.ws.AuthenticationService;
import oracle.workspaces.ws.TemplateService;
import oracle.workspaces.ws.TemplateServiceServiceLocator;
import oracle.workspaces.ws.TemplateServiceSoapBindingStub;
import oracle.workspaces.ws.beans.WorkspaceTemplateItem;
import oracle.workspaces.ws.exceptions.CwWSEException;
import org.apache.axis.transport.http.HTTPConstants;

public class TemplateServiceSample
{
    public static TemplateService configureTemplateService(
        String szAuthCookie) throws ServiceException
    {
        TemplateServiceServiceLocator tssl =
            new TemplateServiceServiceLocator();
        TemplateService tService =
            tssl.getTemplateService();
        ((TemplateServiceSoapBindingStub)tService).
            setMaintainSession(true);
        ((javax.xml.rpc.Stub) tService).
            _setProperty(HTTPConstants.HEADER_COOKIE, szAuthCookie);
        return tService;
    }

    public static WorkspaceTemplateItem[] listTemplates(
        TemplateService szTemplateService)
        throws RemoteException, CwWSEException
```

```
{  
  
    // Retrieve all workspace templates stored in Oracle  
    // Workspaces, and output information about each of them  
  
    WorkspaceTemplateItem[] wspcItem =  
        szTemplateService.listWorkspaceTemplates();  
  
    if (wspcItem == null) {  
        System.out.println("No templates were found");  
    } else  
    {  
        for (int i =0; i < wspcItem.length; i++)  
        {  
            System.out.println("Template #" + i + ":");  
            System.out.println("Created by: " + wspcItem[i].getTemplateCreatedBy());  
            System.out.println("Name: " + wspcItem[i].getTemplateName());  
            System.out.println("Web UI URI: " + wspcItem[i].getWebUIUrl());  
        }  
    }  
    return wspcItem;  
}  
  
public static void main(String[] args)  
{  
    try {  
        // Get authentication cookie  
  
        AuthenticationService myAuthenticationService =  
            AuthenticationSample.configureAuthenticationService(  
                "rg4",  
                "welcome1");  
  
        String authCookie = AuthenticationSample.getAuthenticationCookie  
            (myAuthenticationService);  
  
        System.out.println("Retrieved authentication cookie : " + authCookie);  
  
        // Get TemplateService and set authentication cookie  
  
        TemplateService myTemplateService =  
            TemplateServiceSample.configureTemplateService(authCookie);  
  
        // Output information about each template  
  
        TemplateServiceSample.listTemplates(myTemplateService);  
  
        myAuthenticationService.logout();  
  
    } catch (CwWSEException cwse)  
    {  
        System.out.println("CwWSEException caught: " + cwse.getMessage());  
        cwse.printStackTrace();  
    } catch (Exception e)  
    {  
        System.out.println("Exception caught: " + e.toString());  
    }  
}  
}
```

---

---

## Administration Service

The Administration service allows you to perform administrative operations in Oracle Workspaces such as granting and revoking application level roles, and setting application lever configuration properties.

The environment property "oracle.workspaces.ws.administrationEnabled" must be set to true.

---

---

**Note:** It is possible to revoke application administrator role from all of your Oracle Workspaces administrators. You may use the Oracle Workspaces Configuration Assistant to grant application administrator role to a particular user.

Ensure that the environment variables ORACLE\_HOME and LD\_LIBRARY\_PATH are defined. Execute the following command at a command prompt:

```
java -jar ${ORACLE_HOME}/workspaces/lib/workspaces_ca.jar
logfile=cw_grant_admin_role.log
action=grant_admin_role
oh=<Infrastructure Oracle home directory>
oid=<Oracle Internet Directory host>
oid_port=<Oracle Internet Directory port>
oid_user_dn=<Oracle Internet Directory administrator's
distinguished name>
oid_passwd=<Oracle Internet Directory administrator's
password>
db_sn=<Oracle Workspaces database entry orclbdbglobalname>
cw_admin_uid=<UID of the user to be granted application
administrator role>
```

---

---

### Example: Grant Roles to Members

The following sample grants application administrator role and workspace creator role to a group of members. These members are created from a group of users who have been defined in Oracle Internet Directory. Before executing the sample, either add users to the group as specified in the sample or specify your own group in Oracle Internet Directory and populate it with users.

**Example 17-1 AdministrationServiceSample.java**

```
package oracle.sample.workspaces.ws;

import java.rmi.RemoteException;
import javax.xml.rpc.ServiceException;
```

```
import oracle.workspaces.ws.AdministrationService;
import oracle.workspaces.ws.AdministrationServiceServiceLocator;
import oracle.workspaces.ws.AdministrationServiceSoapBindingStub;
import oracle.workspaces.ws.AuthenticationService;
import oracle.workspaces.ws.UsersService;
import oracle.workspaces.ws.beans.Member;
import oracle.workspaces.ws.beans.User;
import oracle.workspaces.ws.exceptions.CwWSEException;
import org.apache.axis.transport.http.HTTPConstants;

public class AdministrationServiceSample
{
    public static AdministrationService configureAdministrationService(
        String szAuthCookie)
        throws ServiceException, RemoteException, CwWSEException
    {
        AdministrationServiceServiceLocator adssl =
            new AdministrationServiceServiceLocator();
        AdministrationService aService =
            adssl.getAdministrationService();
        ((AdministrationServiceSoapBindingStub)aService).
            setMaintainSession(true);
        ((javax.xml.rpc.Stub)aService).
            _setProperty(HTTPConstants.HEADER_COOKIE, szAuthCookie);

        ((AdministrationServiceSoapBindingStub)aService).
            setSystemAdministratorMode(true);

        return aService;
    }

    public static void grantRolesToUsers(
        AdministrationService szAdministrationService,
        User[] szUsers,
        String[] szRoles)
        throws RemoteException, CwWSEException
    {
        // The last argument of the method updateApplicationRoles
        // is an array of strings. Each string in this
        // array may have a value of either "GRANT" or
        // "REVOKE". The array must be the same
        // size as the number of members to whom you wish to
        // grant or revoke application roles.
        //
        // The following for loop sets each string in
        // grantStrings to the value "GRANT". As a result,
        // each member in currentMembers will be granted
        // the specified application role (which is specified
        // by szRoles).

        String[] grantStrings = new String[szUsers.length];
        for (int i=0; i < grantStrings.length; i++)
        {
            grantStrings[i] = "GRANT";
        }

        Member[] currentMembers = new Member[szUsers.length];
```

```

    for (int i=0; i < szUsers.length; i++)
    {
        Member currentMember = new Member();
        currentMember.setMemberRoleType(szRoles);
        currentMember.setUser(szUsers[i]);
        currentMembers[i] = currentMember;
    }

    szAdministrationService.updateApplicationRoles(
        currentMembers, grantStrings);
}

public static void main(String[] args)
{
    try {
        // Get authentication cookie

        AuthenticationService myAuthenticationService =
            AuthenticationSample.configureAuthenticationService(
                "orcladmin",
                "welcome1");

        String authCookie = AuthenticationSample.getAuthenticationCookie
            (myAuthenticationService);

        System.out.println("Retrieved authentication cookie : " + authCookie);

        // Get UsersService and set authentication cookie
        UsersService myUsersService =
            UsersServiceSample.configureUsersService(authCookie);
        // Retrieve AdministrationService and set authentication cookie
        AdministrationService myAdministrationService =
            AdministrationServiceSample.configureAdministrationService(authCookie);

        // Retrieve Oracle Collaboration Suite users
        // from Oracle Internet Directory from
        // the specified group DN
        User[] myUsers = UsersServiceSample.getUsersFromGroupDN(
            myUsersService,
            "cn=raymond_group,cn=Groups,dc=us,dc=oracle,dc=com");

        System.out.println("Retrieved users");

        AdministrationServiceSample.grantRolesToUsers(
            myAdministrationService,
            myUsers,
            new String[]{"APPLICATION_ADMINS_ROLE", "WORKSPACE_CREATORS_ROLE"});

        System.out.println("Granted application administrator " +
            "and workspace creators roles to users");

        myAuthenticationService.logout();
    } catch (CwWSEException cwse)
    {
        System.out.println("CwWSEException caught: " + cwse.getMessage());
    } catch (Exception e)
    {

```

```
        System.out.println("Exception caught: " + e.toString());
    }

}
}
```

---

---

## Oracle Workspaces Web Services Performance Tips

The following are some factors that affect the performance of Oracle Workspaces Web Services:

- All the bulk operation calls take relatively more time because they have to create or retrieve bulk data. The time required for such an operation will be proportional to the number of operations being attempted in a single bulk operation call.
- Cross-workspace search is slower if the number of workspaces of which a user is a member is extremely large. In general, search operations take more time if the wild card search criterion (\* or %) is given.
- The operation to list all workspaces or list the user's favorite workspaces will take more time if there are any attributes being requested through the attrRequest parameter.
- Trying to retrieve a large number of items through a list or get call will take more time.
- Creation or deletion of a workspace inherently takes more time than other operations.
- Creation or deletion of a resource in a workspace may take more time.
- Oracle Workspaces Web services operations may result in a request timeout or client socket timeout exception if the data being created or retrieved is huge. In case of operations where the data is being retrieved, the user will have to narrow down or refine his or her calls to retrieve a subset of the data he or she originally intended to retrieve. In case of create, update, or delete operations throwing a timeout exception, these operations would still go through unless there is an error in user input. The user will not get any notifications either way.



---

---

# Index

## A

---

Administration service, 17-1  
Announcements service, 11-1  
Apache Axis, 2-1  
application administrator role  
    grant to user, 17-1  
architecture, 1-2  
Attachments service  
    Web services  
        Attachments service, 15-1  
authentication cookies  
    retrieving with authentication service, 3-1  
    retrieving without S2S Authentication  
        service, 4-3  
Authentication service, 3-1

## B

---

building applications, 2-3

## C

---

capabilities, 1-1  
classpath, 2-1  
client-side stub classes  
    compiling, 2-3  
    generating, 2-2  
    generating with Apache Ant, 2-2

## D

---

digest authentication, using with S2S Authentication  
    service, 4-2  
Discussion service, 12-1

## E

---

environment variables  
    setting, 4-1  
examples  
    add members and resources to a workspace, 7-1  
    create and search for tasks, 10-1  
    create announcement, 11-1  
    create attachment, 15-1  
    create board, post message, and post reply, sample  
        code

DiscussionServiceSample.java, 12-1  
create folders in and retrieve contents of  
    library, 8-1  
create meeting, 9-1  
create new workspace, 5-1  
create view, add items to view, and search for  
    views that contain item, 14-1  
grant roles to members, 17-1  
post message to inbox, 13-1  
retrieve authentication cookie, 3-1  
retrieve templates, 16-1  
retrieve users, 6-1

## H

---

Home service, 5-1

## I

---

Inbox Service, 13-1

## L

---

Library service, 8-1

## M

---

Meetings service, 9-1

## O

---

Oracle Workspaces Configuration Assistant, 17-1

## P

---

partner applications  
    registering with Oracle Internet Directory, 4-1  
performance tips, A-1

## R

---

requirements, 2-1

## S

---

sample code  
    AdministrationServiceSample.java, 17-1

- AnnouncementsServiceSample.java, 11-1
- AttachmentsServiceSample.java, 15-1
- AuthenticationSample.java, 3-1
- HomeServiceSample.java, 5-1
- InboxServiceSample.java, 13-1
- LibraryServiceSample.java, 8-1
- MeetingsServiceSample.java, 9-1
- S2SClientSample.java, 4-3
- TasksServiceSample.java, 10-1
- TemplateServiceSample.java, 16-1
- UsersServiceSample.java, 6-1
- ViewsServiceSample.java, 14-1
- WorkspacesServiceSample.java, 7-1

Service-to-Service Authentication service, 4-1

stub classes

- compiling, 2-3
- generating, 2-2
- generating with Apache Ant, 2-2

## **T**

---

- Tasks service, 10-1
- Template service
  - Web services
    - Template service, 16-1

## **U**

---

- Users service, 6-1

## **V**

---

- Views service, 14-1

## **W**

---

Web services

- Administration service, 17-1
- Announcements service, 11-1
- Authentication service, 3-1
- Discussion service, 12-1
- Home service, 5-1
- Inbox service, 13-1
- Library service, 8-1
- Meetings service, 9-1
- Service-to-Service Authentication service, 4-1
- Tasks service, 10-1
- Users service, 6-1
- Views service, 14-1
- Workspaces service, 7-1

Workspaces service, 7-1

WSDL files

- location, 2-3