ArcIMS® 9

Customizing ArcIMS—Using the Java Connector

ESRI ArcGIS™

GIS by ESRI™

Contents

Introducing the Java Connector

1

IN THIS CHAPTER

- **ArcIMS Application Server Connectors**
- **What is the ArcIMS—Java Connector?**
- **Why use the Java Connector?**
- **Available resources**

ESRI® ArcIMS® software provides a suite of tools that allows you to create effective Web sites for your mapping and geographic information system (GIS) needs. ArcIMS provides the foundation for the graphical and functional components of these Web sites. You can build on this foundation through customization of the ArcIMS viewers.

Customizing ArcIMS is a series of books that describe customizing the HTML and Java™ Viewers and creating viewers or applications supported by the Java, ActiveX®, and ColdFusion® Connectors.

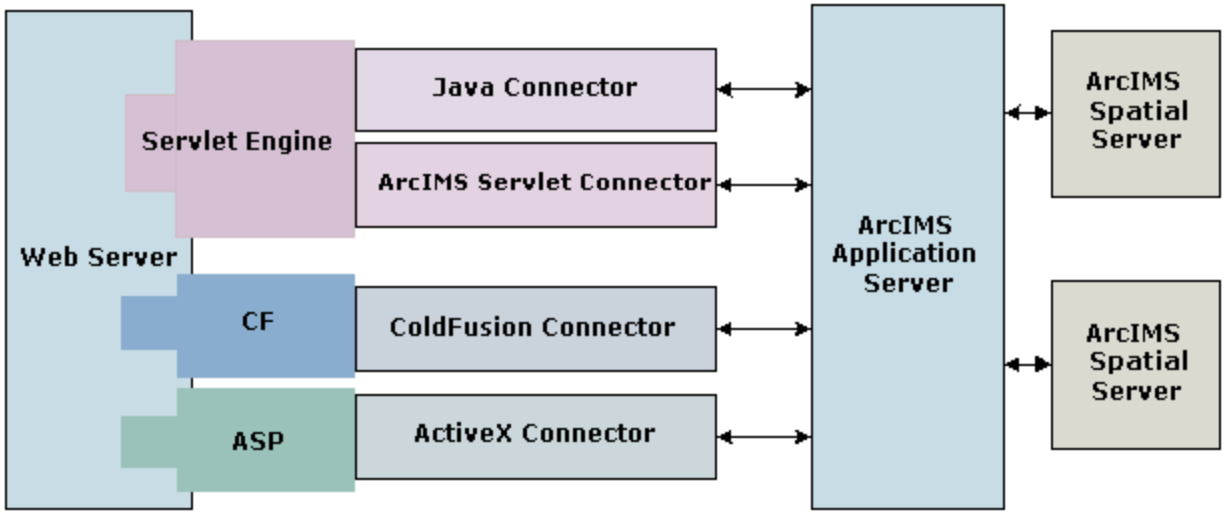
This book explains the ArcIMS—Java Connector and provides a complete reference to the Java Connector Object Model and Java Connector Tag Library.

This book assumes you have a working knowledge of JavaServer Pages™ (JSP™) technology, the Java language, and HTML.

ArcIMS Application Server Connectors

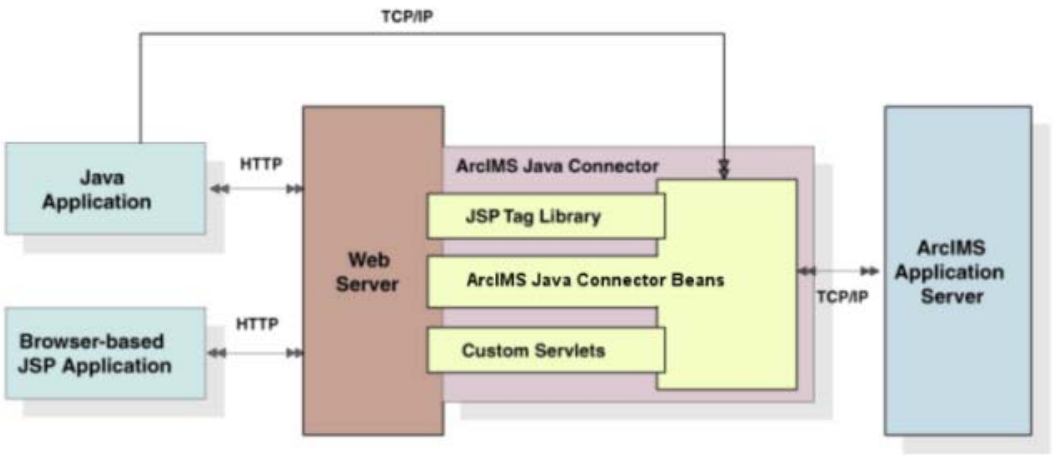
The ArcIMS Application Server Connectors connect the Web server to the ArcIMS Application Server. The ArcIMS Servlet Connector is the standard connector used with ArcIMS; however, there are several alternatives—Java, ColdFusion, ActiveX, and Z39.50 Connectors. You can select a suitable Application Server Connector based on your development environment. The connectors reside on the Web server computer. Each Web server can have many Application Server Connectors as long as they are different types.

The Java Connector communicates with the ArcIMS Application Server via a JSP client or a standalone Java application. The ColdFusion and ActiveX Connectors work with custom clients and translate their own languages into ArcXML. For more information about the ActiveX and ColdFusion Connectors, see their respective *Customizing ArcIMS* books.



What is the ArcIMS—Java Connector?

ArcIMS—Java Connector is an ArcIMS Application Server Connector that allows communication between the ArcIMS Application Server and a JSP client or a standalone Java application. It is a reusable software component suite that includes a JavaBeans™ Object Model Library and a rich set of custom JSP tags supported in the form of a Tag Library.



How does the connector work?

These JavaBeans and JSP tags allow you to programmatically establish communication with an ArcIMS Application Server—via an HTTP, HTTPS, or TCP/IP connection—and begin sending ArcXML requests to it. Once the Application Server receives the request, it processes it and returns the appropriate response. With that response, your application can react accordingly.

For example, suppose you've created a simple data browsing application that allows a user to pan and zoom around a map. To zoom in, the user pushes a button on your application's user interface. Your application translates that button press as an ArcXML request that gets sent via the Java Connector to the Application Server, which in turn passes it to the Spatial Server for processing. Based on this request, the Spatial Server generates a new zoomed-in display of the area of interest and informs the Application Server that the request has been processed. Your application can then display the new image back to the user.

JavaBeans and JSP tags compared

You can customize your Java Connector for ArcIMS using one of the following methods:

- Using JavaBeans (the object model)
- Using JSP tags (the tag library, which exposes the object model)
- A combination of the two, such as using JavaBeans to enforce the business logic while using JSP tags for the presentation

All products on the market today share the same advantages and disadvantages of using JavaBeans versus JSP tags. The main advantage of using JSP tags is the ease of use. JSP tags are easy

to learn and use, especially for nondevelopers, because they work much the same way HTML tags do. In the same way you type HTML tags and attributes into a .html text file, you type JSP tags and attributes into a .jsp file. Just as you use an HTML tag library as a reference to create the .html page, you use a JSP tag library as a reference to create the .jsp page.

While JavaBeans are difficult for nondevelopers to learn and use, they give more flexibility and functionality than JSP tags and can sometimes simplify processes. While JSP tags can be used only with JSP pages, JavaBeans can be used in any Java program and JSP pages.

Java Connector Object Model

The Java Connector Object Model is a collection of server-side JavaBeans that implement the ArcXML specification. You can utilize the object model beans and their methods in your application to implement map display functions; perform rendering and symbology; add dynamic layers; and perform feature and spatial queries, address geocoding, projections, and metadata functions. For more information, see Chapter 3, 'Java Connector Object Model'.

Java Connector Tag Library

The Java Connector Tag Library is a collection of custom JSP tags built on the Java Connector Object Model. These tags provide high-level access to the entire server-side Object Model library. Web developers with little or no knowledge of Java can now develop advanced mapping applications or integrate mapping capabilities into existing applications using the JSP Tag Library.

Applications developed using the JSP architecture encapsulate the business logic—such as database access, security, and transaction integrity—and isolate it from the underlying complexity. For more information about the Tag Library and the various tags supported, see Chapter 4, 'Java Connector Tag Library'.

Features of the connector

- Communicates with the ArcIMS Application Server via an HTTP, HTTPS, or TCP/IP connection.
- Includes a rich collection of custom JSP tags in the form of a Tag Library.
- Includes server-side, object model JavaBeans for mapping, display, query, and analysis.
- Supports user authentication so that you can control access to your sensitive data.
- Provides the ability to overlay local and remote data.
- Provides complete support for all ArcXML capabilities that can be processed on the server-side.
- Supports an open, cross-platform architecture. Web servers, platforms, and other components can be easily upgraded or switched without affecting your applications.

Why use the Java Connector?

If you are using Java or JSP to create a custom application and you want to incorporate geographic data in it, you can use the Java Connector to connect your application directly to the ArcIMS Application Server. The Application Server processes client requests sent to it and, if necessary, routes those requests to the appropriate Spatial Server for further processing. By connecting your application to the Application Server, you connect to all the functionality provided by the Spatial Server—such as the ability to view and query your geographic data. Thus, you don't have to write your application from scratch; instead, you can rely on ArcIMS—through the Java Connector—to provide the geographic operations you need.

- Use JSP to build a browser-based, thin Web application that allows people outside your organization to search or browse for information that contains geographic content. The application logic may reside in server-based resources, such as JavaBeans, to provide additional Internet security.
- Update or create an interactive Web site with mapping capabilities—such as address geocoding, feature selection using spatial filters, and selective visibility of layers based on scale—even if you're a Web developer with little or no knowledge of the Java language.
- Create a standalone Java application that allows people within or outside your organization to browse maps.
- Design a custom servlet for an Internet mapping application where the custom business logic resides behind a Web server.

Available resources

Helpful documentation included with ArcIMS

- This book, *Customizing ArcIMS—Java Connector*, introduces you to ArcIMS—Java Connector and its capabilities. Chapter 2 contains a tutorial and is usually the best place to start. Chapter 3 describes the Java Connector Object Model. Chapter 4 introduces you to the JSP Tag Library and provides an alphabetical listing of all tags. The appendix explains how you can migrate your AppServerLink applications to Java Connector.
- For information on installing the ArcIMS—Java Connector, see the *ArcIMS Installation Guide*.
- Detailed information on the methods and classes in the ArcIMS—Java Connector Object Model is provided in the javadoc-generated HTML reference *ArcIMS—Java Connector Object Model API Specification*.
- The *ArcXML Programmer's Reference Guide* explains how to customize map configuration files.
- The Java Connector includes a number of sample applications. Once installed, steps for configuring and running the samples are available in <ArcIMS Installation Directory>/ArcIMS/Samples/Java/readme_samples.htm.

The ArcIMS Knowledge Base

The ArcIMS Knowledge Base is a technical support database containing:

- Frequently asked questions
- How-to instructions
- Troubleshooting tips
- Error messages

You can search the Knowledge Base by typing keywords or browsing through folders. You can access the ArcIMS Knowledge Base through ESRI Online Support Center by typing http://support.esri.com as the Web address.

Contacting ESRI

If you need to contact ESRI for technical support, see the support card you received with ArcIMS or refer to 'Technical support' under 'Getting started' in the 'Introduction' chapter of *ArcIMS Help*. You can also visit ESRI on the Web at www.esri.com.

ESRI education solutions

ESRI provides educational opportunities related to geographic information science, GIS applications, and GIS technology. You can complete instructor-led courses and Web-based courses or work through self-study workbooks. ESRI has education solutions fitting every learning style and pocketbook. For more information, go to www.esri.com.

Getting started

2

IN THIS CHAPTER

- **Installing the Java Connector**
- **Java Connector samples**
- **Quick-start tutorial**

This chapter will help you get started with your application development using the Java Connector.

The tutorial included later in this chapter moves you through the process of creating a map and deploying it as a Web application. The application can be generated by using either the beans from the Java Connector Object Model or the JSP tags from the Tag Library.

You will need the following additional documents to get started:

- *ArcIMS Installation Guide*—for information on installing the Java Connector.
- *Java Connector Samples Setup*—for details on configuring the Java Connector samples once they're installed. This file is located at <ArcIMS Installation Directory>/ArcIMS/Samples/Java/readme_samples.htm.

Installing the Java Connector

You can install the Java Connector and associated samples by selecting them during the ArcIMS installation dialog box or script. On Windows and Solaris, you can also choose to install Java Connector-related documentation. On all platforms, documentation can be copied from the CD–ROM to your local system if desired.

For step-by-step instructions on installing the Java Connector, see the section 'Step 3: Install ArcIMS: ArcIMS custom Application Server Connectors' in the *ArcIMS Installation Guide*.

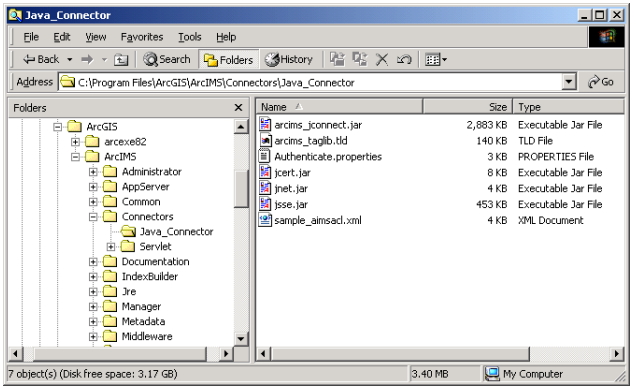
Overview of installed components

Once installed, the various components of the Java Connector are available in the following installation directory:

Windows—<ArcIMS Installation Directory>\ArcIMS\Connectors\Java_Connector

UNIX/Linux—<ArcIMS Installation Directory>/ArcIMS/Middleware/Java_Connector

Based on the options selected during installation, additional documents and samples will be installed into the <ArcIMS Installation Directory>/ArcIMS/Documentation and <ArcIMS Installation Directory>/ArcIMS/Samples folders, respectively.



The key installed files are:

arcims_jconnect.jar—contains the object model Java class files

arcims_taglib.tld—the Tag Library Descriptor (TLD) file

Authenticate.properties—the Java Connector authentication file

jsse.jar—dependent Java Archive (JAR) file provided by Sun Microsystems, Inc.

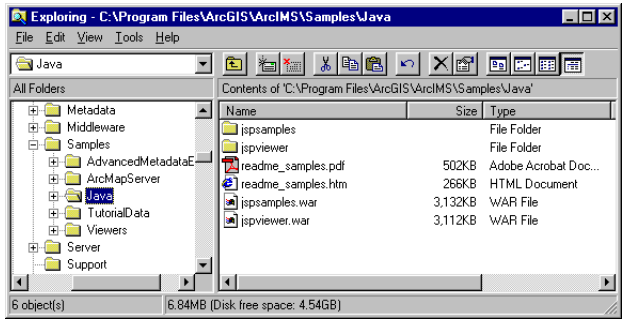
jnet.jar—dependent JAR file provided by Sun Microsystems, Inc.

jcert.jar—dependent JAR file provided by Sun Microsystems, Inc.

sample_aimsacl.xml—a sample Access Control List (ACL) file that can be renamed and used for authentication

Java Connector samples

The Java Connector samples are designed to introduce you to the various aspects and functionality supported by the Java Connector Object Model and Tag Library. If selected, the install program copies the Java Connector samples into the <ArcIMS Installation Directory>/ArcIMS/Samples/Java folder.



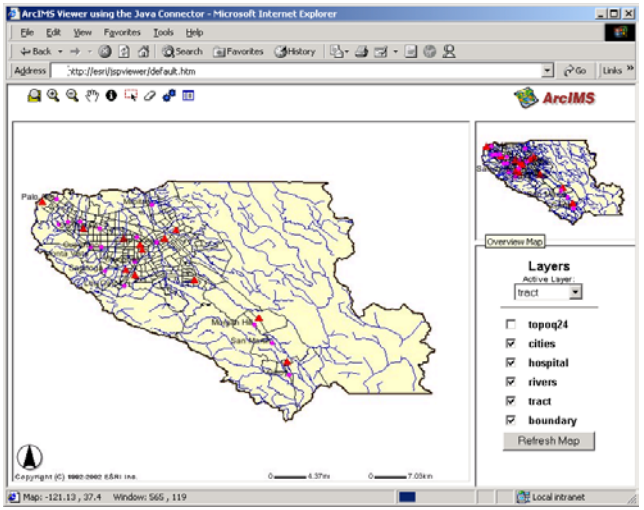
These samples demonstrate specific functionality using the object model beans and the JSP tags in the Tag Library. You can learn more about using the beans and tags by modifying the samples and adding additional functionality.

You can install the Java Connector samples as separate Web applications or as individual files. The advantage of configuring JSP as a Web application is that it is self-contained—all the dependent files are in one location. If you choose to configure JSP as individual files, you must add the dependent files to your class path. For ease of deployment, the Java Connector samples were designed to run as Web applications.

For detailed information on how to configure the samples, see *Java Connector Samples Setup* located at <ArcIMS Installation Directory>/ArcIMS/Samples/Java/readme_samples.htm.

Java Connector sample viewer

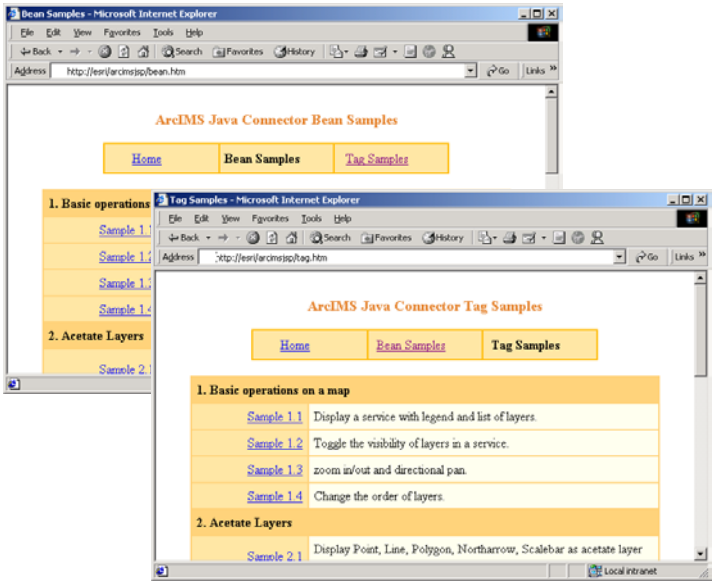
The Java Connector viewer sample is available from <ArcIMS Installation Directory>/ArcIMS/Samples/Java/jspviewer. It is a template application that can be customized and used as a springboard for development of more complex Java Connector applications.



Additional Java Connector Samples

In addition to the sample viewer, various smaller samples are available from <ArcIMS Installation Directory>/ArcIMS/Samples/Java/jspsamples. View a list of these samples by opening the index.htm file. Their source code is contained in the subfolder 'src'.

Two versions of each sample were developed—one that uses the Java Connector Object Model beans and a second that uses the JSP tags in the Java Connector Tag Library.



These samples demonstrate the following:

Sample 1. Basic operations in a Map

Sample 1.1: Displays a service with legend and list of layers.

Sample 1.2: Toggles the visibility of layers in a service.

Sample 1.3: Zooms in/out and performs directional pan.

Sample 1.4: Changes the order of layers.

Sample 2. Acetate layers

Sample 2.1: Displays a point, line, polygon, North arrow, and scalebar as an acetate layer on a map.

Sample 3. Dynamic data

Sample 3.1: Adds data from a shape workspace.

Sample 3.2: Adds data from an image workspace.

Sample 3.3: Adds data from an ArcSDE® workspace.

Sample 4. Query

Sample 4.1: Selects features with buffer as spatial filter.

Sample 4.2: Uses a line to select features.

Sample 4.3: Identifies features in a layer.

Sample 4.4: Buffers an envelope to select features in a layer.

Sample 4.5: Extracts the features of a layer inside an envelope.

Sample 5. Renderers and symbols

Sample 5.1: Demonstrates a simple Label Renderer.

Sample 5.2: Demonstrates a simple Line Symbol.

Sample 5.3: Demonstrates a Gradient Fill Symbol.

Sample 5.4: Demonstrates a Hash Line Symbol.

Sample 5.5: Demonstrates a Value Map Renderer (range).

Sample 5.6: Demonstrates a Value Map Renderer (exact).

Sample 6. Image streaming

Sample 6.1: Displays an image output as stream.

Sample 7. ArcMap service

Sample 7.1: Displays an ArcMap™ service.

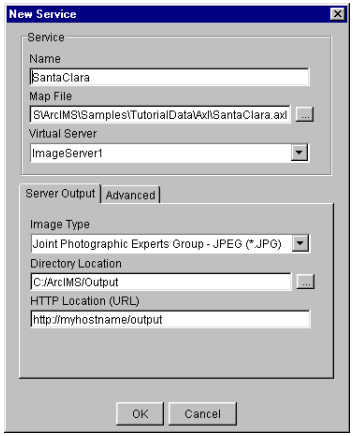
Quick-start tutorial

This tutorial introduces the key steps involved in creating a Web application using the Java Connector. You will learn how to:

- Set up your development environment.
- Create a Web application descriptor file—web.xml.
- Create a Web application by using the object model beans or the Tag Library.
- Deploy your Web application.
- Display your map.

Prerequisite: Create an Image Service named SantaClara

Before proceeding with the tutorial, create an Image Service named SantaClara. Use SantaClara.axl (<ArcIMS Installation Directory>/ArcIMS/Samples/TutorialData/Axl) as the service's map configuration file. For instructions on creating a service, see the ArcIMS online Help.



Exercise 1: Set up your development environment

Set up the development environment before you begin. This tutorial will consist of an unpacked Web application.

1. Create a new directory named jsptutorial.
2. Create a subdirectory named source under /jsptutorial.

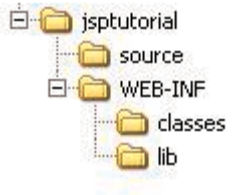
   Content, such as your application's JSP and HTML pages, text files, and images, is stored in this directory.
3. Create an additional subdirectory named WEB-INF under /jsptutorial.

   The container looks in this directory for deployment information about the Web application.
4. Create a subdirectory named classes under /jsptutorial/WEB-INF.

   The container looks in this directory for classes used in the Web application.
5. Create an additional subdirectory named lib under /jsptutorial/WEB-INF.

   The container looks in this directory for class libraries used in the Web application.

Once complete, the file structure that holds the tutorial Web application should look like this:



Exercise 2: Create a Web application descriptor file—web.xml

In order for the Web server or servlet engine's JSP container to understand the tutorial application, you need to describe it. This is done in a Web application descriptor file—web.xml—which resides in the /WEB-INF directory of the application. This file holds configuration information about your Web application.

In this section, you will create web.xml, set it up with minimum configuration settings, and copy it into the WEB-INF directory. Additional details on configuration settings are available from http://java.sun.com/j2ee/dtds/web-app_2_2.dtd.

1. Using a text editor, create a new file—web.xml—and type

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   ```

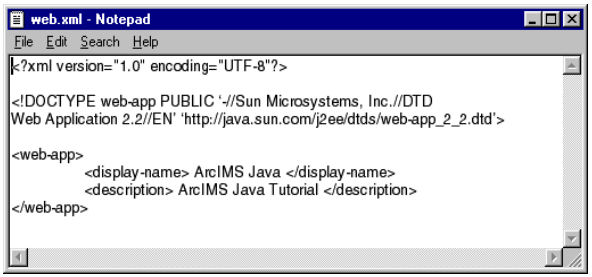
   ```
   <!DOCTYPE web-app PUBLIC '-//Sun Microsystems,
   Inc.//DTD Web Application 2.2//EN' 'http://
   java.sun.com/j2ee/dtds/web-app_2_2.dtd'>
   ```

   The first line defines this file as a version 1.1 XML file. The second line defines the format of the XML to conform to the Web Application 2.2 definitions and indicates where this definition can be found.

2. Type the following below the previous entry:

   ```
   <web-app>
     <display-name> ArcIMS Java </display-name>
     <description> ArcIMS Java Tutorial </description>
   </web-app>
   ```

   The lines above define the display name and description of the Web application. Most Web servers and servlet engines display this information on deployment.



3. Save web.xml to the /jsptutorial/WEB-INF/ directory and close the file.

Exercise 3: Create a Web application

In this section, you will create your Web application using either the Java Connector Object Model beans or the custom JSP Tag Library.

Follow the steps for the development method you prefer.

Method A: Create a Web application using the object model

The Java Connector Object Model is contained within a JAR file—arcims_jconnect.jar. Three additional dependent JAR files—jsse.jar, jnet.jar, and jcert.jar—are provided by Sun Microsystems, Inc. The Java Connector uses 'xerces 1.3.1' for parsing XML.

For detailed information on the methods and classes in the Java Connector Object Model, see the javadoc-generated HTML reference *ArcIMS—Java Connector Object Model API Specification*.

1. Copy the arcims_jconnect.jar, jsse.jar, jnet.jar, jcert.jar files from the /Java_Connector directory to the /jsptutorial/WEB-INF/lib directory.

2. Using a text editor or a Java Integrated Development Environment (IDE), create a new file—map.jsp—in the /jsptutorial/source/ directory.

   This is your first JSP page using the Java Connector.

3. Type the following code into this file:

```
<jsp:useBean id="connection"
class="com.esri.aims.mtier.io.ConnectionProxy"
scope="page"/>

<jsp:useBean id="map"
class="com.esri.aims.mtier.model.map.Map"
scope="page"/>
```

   The tag <jsp:useBean> instantiates an object identifier. In the above code, the objects *connection* of the ConnectionProxy class and *map* of the Map class were instantiated. The scope of these objects was set as page.

4. Continuing in the map.jsp file, below the previous entry type:

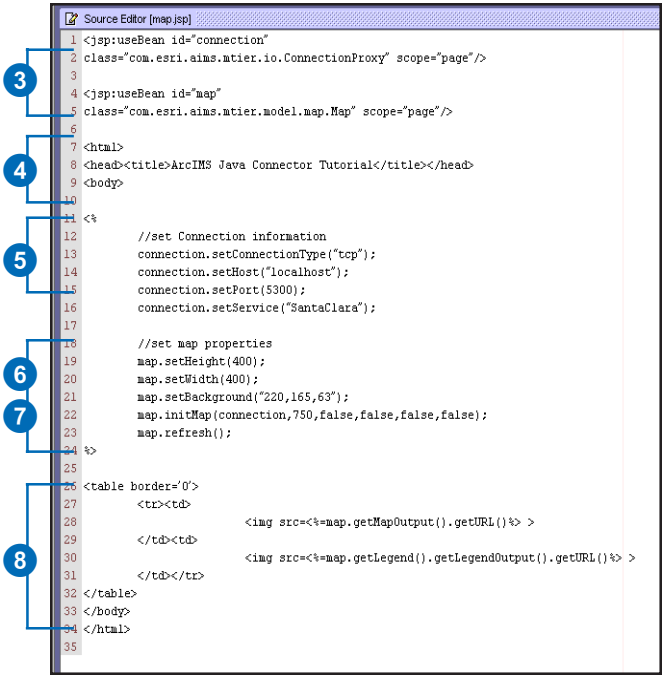
```
<html>
<head><title>ArcIMS Java Connector Tutorial
</title></head>
<body>
```

   This code starts an HTML portion of the JSP file, sets its title, and opens a <body> tag.

5. The entry point into the object model is the ConnectionProxy object that was instantiated in Step 3—connection. Continuing below the existing entries in the map.jsp file, type:

```
<%
   //set Connection information
   connection.setConnectionType("tcp");
   connection.setHost("localhost");
   connection.setPort(5300);
   connection.setService("SantaClara");
```

   This code sets connection's type to TCP, host to localhost, port to 5300, and service to SantaClara.



6. Set properties for the other object instantiated in Step 3—map. Add the following below the previous text in map.jsp:

```
//set map properties
map.setHeight(400);
map.setWidth(400);
map.setBackground("220,165,63");
```

   The background color is set in RGB values. Height and width properties use pixel values. Other properties to the Map object are available and can be set as needed.

7. Continuing, type:

```
    map.initMap(connection,750,false,false,
            false,false);
    map.refresh();
%>
```

   This initializes map using the connection previously established in Step 5. The map is initialized at 750 dpi, and envelope, renderers, recordset, and extensions are not initialized. The map is refreshed to create a URL output with the latest properties of the Map object.

8. Below the previous entry, type:

```
<table border='0'>
 <tr><td>
    <img src=<%=map.getMapOutput().getURL()%> >
 </td><td>
    <img src=<%=map.getLegend().getLegendOutput
                                    ().getURL()%> >
 </td></tr>
</table>
</body>
</html>
```

   The URL output generated by refreshing map in Step 7 is used to display the map and legend in the application.

9. Save and close the file.

You can now deploy the application by continuing on to 'Exercise 4: Deploy a Web application' of this tutorial.

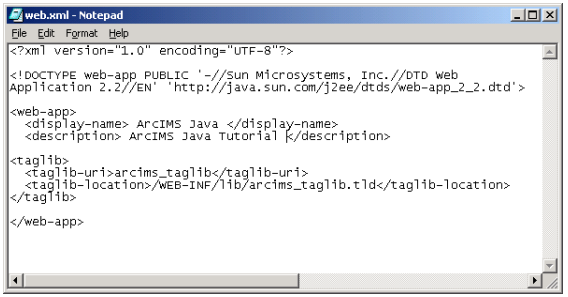
Method B: Create a Web application using the Tag Library

The Java Connector Tag Library is a collection of custom JSP tag elements. The Tag Library Descriptor file is named arcims_taglib.tld.

1. Copy the arcims_taglib.tld file from the /Java_Connector directory to the /jsptutorial/WEB-INF/lib directory.

2. Using a text editor, open /jsptutorial/WEB-INF/web.xml and add the following lines above the closing tag </web-app>:

```
<taglib>
 <taglib-uri>arcims_taglib</taglib-uri>
 <taglib-location>/WEB-INF/lib/
            arcims_taglib.tld</taglib-location>
</taglib>
```



   JSP files access the Tag Library using the name arcims_taglib. The tag <taglib-location> specifies the actual location of the TLD file.

3. Save and close web.xml.

4. Using a text editor or a Java IDE, create a new file—maptag.jsp—in the /jsptutorial/source/ directory.

   This is your first JSP using the Tag Library.

5. Type the following code into this file:

```
<%@taglib uri="arcims_taglib" prefix="aims" %>
```

   The Tag Library has to be referenced in order for the JSP file to utilize it. The *uri* is the name specified in the web.xml file in Step 2. The *prefix* fully qualifies the tags used. Specifying it helps to avoid conflicts if more than one tag library is being used inside the JSP file.

6. Below the previous entry, add

```
<html>
<head><title>ArcIMS Java Tag Library
                    Tutorial</title></head>
<body>
```

This code starts an HTML portion of the JSP file, sets its title, and opens a <body> tag.

7. Create an HTTP connection object and an ArcIMS service object by adding the following lines:

```
<aims:httpConnection id="myhttp" url="http://
                                  localhost" />
<aims:mapService id="santaclara"
                  connectionId="<%=myhttp%>"
                       name="SantaClara" />
```

8. Continue by adding the following:

```
<aims:map id="mymap" legend="mylegend"
    serviceId="<%=santaclara%>" width="500"
                                height="500" />
```

The above line creates a map object by passing the service object created in Step 7 as its parameter. The service object holds the connection information.

9. Continuing in the maptag.jsp file, below the previous entry type:

```
<table border='0' >
  <tr><td>
    <img src=<%=mymap%> >
  </td><td>
    <img src=<%=mylegend%> >
  </td></tr>
</table>
</body>
</html>
```

This sets the map variable identifier and legend holding the URL path used to display the map and legend in the application.

10. Save and close maptag.jsp.

You can now deploy the application by continuing on to 'Exercise 4: Deploy a Web application' of this tutorial.

Exercise 4: Deploy a Web application

In this section you will deploy the Web application that you've created. Deployment, for most configurations, involves creating a virtual directory that points to the file location of the application and assigning a name to that directory and to the Web application itself. While you can use any name, *jsptutorial* was used as the associated name.

Instructions for deploying a Web application are dependent on the particular platform, Web server, and servlet engine combination. Steps for a few different options are given in this exercise.

If your combination is not shown here,

- Metadata Explorer and ArcIMS Service Administrator are Web applications developed using the Java Connector. *ArcIMS Installation Guide* includes

deployment instructions for these applications on a large number of different Web server/servlet engine combinations. In addition, the setup instructions for the Java Connector samples—also deployed Web applications—provide directions for a number of configurations.

Use the 'Configuring your Web server' steps provided in these documents to walk you through the deployment process. However, remember to revise any references to Web application names, virtual directory names, and pathnames to correspond with this tutorial.

ArcIMS Installation Guide can be accessed from the ArcIMS CD–ROM. *Java Connector Samples Setup* is available at <ArcIMS Installation Directory>/ArcIMS/Samples/Java/readme_samples.htm if the Java Connector samples have been installed.

or,

- Refer to your Web server and servlet engine documentation for Web application deployment instructions.

Note: Some Web server and servlet engines require the application to be packaged into a Web archive (WAR) file before deployment. This process is not documented here. Refer to your Web server or servlet engine documentation for details.

Apache 1.3.x and Tomcat 3.2.x (UNIX and Windows)

1. Copy the folder /jsptutorial into <Jakarta-Tomcat Installation Directory>/webapps directory.
2. Restart both Apache and Tomcat.

IIS and ServletExec 3.1 or 4.0 (Windows only)

1. Verify that ServletExec is configured for JSP by doing the following:
   a. Open ServletExec via http://localhost/servlet/admin.
   b. In the left panel under Servlets, click Configure and verify that JSP10Servlet has been loaded.
   c. In the left panel under Servlets, click Aliases and verify that there is an entry for *.jsp that points to JSP10Servlet.
2. Open ServletExec Admin.
3. In the left panel under Web Applications, click Configure.
4. Click Add Web Application.
5. Choose an Application Name, such as ArcIMS Tutorial.
6. Enter a name for the URL Context Path. This is the name you will use to run the samples. For example, /jsptutorial/.

   Note: The context path—/jsptutorial/ in this case—is case sensitive. If you want to access jsptutorial with mixed or upper case names, you might want to add more than one Web application with context pathnames, such as JSPTutorial or JSPTUTORIAL, all pointing to the same location: <full path to the jsptutorial directory>.
7. Enter the full path to jsptutorial directory under Location.
8. Restart both IIS and ServletExec.

IIS and Tomcat 3.2.x (Windows only)

1. Copy the folder /jsptutorial into <Jakarta-Tomcat Installation Directory>/webapps directory.
2. In the IIS Console, add a new virtual directory called jsptutorial.

   To add virtual directories to IIS:

   a. Open the IIS Internet Service Manager.

   b. Under your localhost name, find the Default Web Site. Click to highlight it.

   c. Click Action in the toolbar.

   d. From the dropdown menu, point to New and click Virtual Directory. The New Virtual Directory wizard displays.

   e. For the alias, type a name, such as jsptutorial, and click Next.

   f. Click Browse to select the directory at <Full path to jsptutorial directory>.

   g. For IIS 4.0, check Allow Directory Browsing and click Finish.

      For IIS 5.0, check Browse and click Next. jsptutorial appears in the virtual directory list.
3. In a text editor, open <Jakarta-Tomcat Installation Directory>\conf\uriworkermap.properties, scroll to the end of the file, and add

   /jsptutorial/*=ajp12
4. Save and close the file.
5. Restart both IIS and Tomcat.

iPlanet 6.0 (UNIX and Windows)

1. Verify that iPlanet has JSP enabled and is pointing to the correct Java 2 SDK compiler by doing the following:

   a. Open iPlanet Web Server Administration Server via http://localhost:<port number>.

      Alternately, if you already have iPlanet's Web Server Administration Server open, select Web Server Administration Server from the Server Manager menu at the top of your browser window.

   b. Click Global Settings from the top navigation bar.

   c. In the left panel, click Configure JRE/JDK Paths.

   d. Choose JDK.

   e. Specify the Java 2 SDK path, for example, /<JDK installation directory>/jdk1.3.x.

      Note: The libpath and classpath can be left blank.

   f. Click OK, click Save and Apply, and click OK.

   g. Click Servers in the top navigation bar.

   h. Click Manage.

   i. Click Java in the top navigation bar and verify that Enable Java Globally is checked.

   j. Click OK, click Save and Apply, and click OK.
2. Open <iPlanet installation directory>\https-<localhost>.<domain>.com\config\web-apps.xml in a text editor.
3. Scroll to the end of file and add the following lines before </vs>:

```
<web-app uri="/jsptutorial" dir="<Full path to
   jsptutorial directory>" />
```

Make sure you specify a valid directory path for web-app. Invalid entries will prevent your iPlanet Web Server from starting.

Note: The uri value—/jsptutorial/ in this case—is case sensitive. If you want to access jsptutorial with mixed or uppercase names, you might want to add more than one Web application with context pathnames, such as JSPTutorial or JSPTUTORIAL, all pointing to the same location: <full path to the jsptutorial directory>.

4. Restart the iPlanet Web Server.

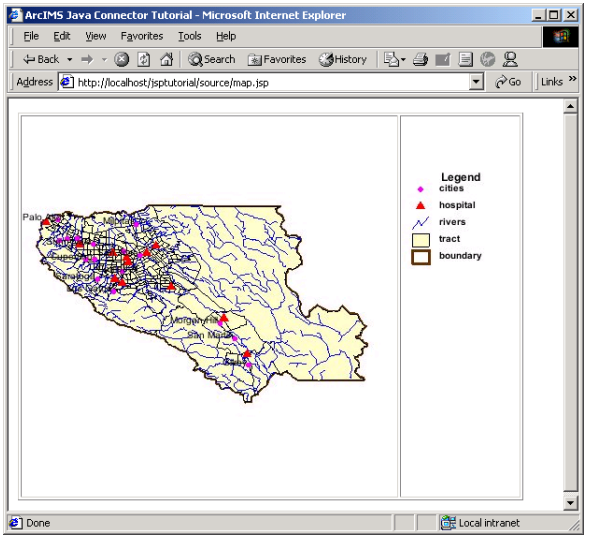
Exercise 5: Display your map

To display the map that you created in this tutorial:

1. Make sure your Web server or servlet engine and ArcIMS are running.
2. To display the Java Connector Object Model bean tutorial, open the URL <http://localhost/jsptutorial/source/map.jsp> in your Web browser.

   To display the Java Connector Tag Library tutorial, open the URL <http://localhost/jsptutorial/source/maptag.jsp> in your Web browser.

Congratulations! You have created your first Web application using the Java Connector.



Java Connector Object Model

3

IN THIS CHAPTER

- **About the API Specification document**
- **Supported development environments**
- **Object model overview**

The ArcIMS—Java Connector Object Model is a collection of server-side JavaBeans that implement the ArcXML specification. You can utilize these beans and their methods in your application to implement map display functions, rendering, and symbology; to add dynamic layers; to perform feature or spatial queries, address geocoding, and projections; and to administer and authenticate services.

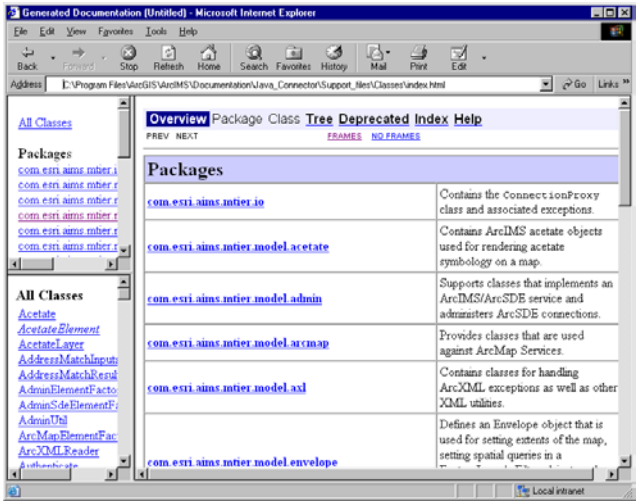
About the API Specification document

The *ArcIMS—Java Connector Object Model API Specification* document is a javadoc-generated HTML reference that is available as a documentation option during a custom install of ArcIMS. The application programming interface (API) Specification document provides complete details on all packages, classes, interfaces, exceptions, and methods that are available in the Java Connector Object Model.

This API document has pages corresponding to the items in the navigation bar, as described below.

Overview

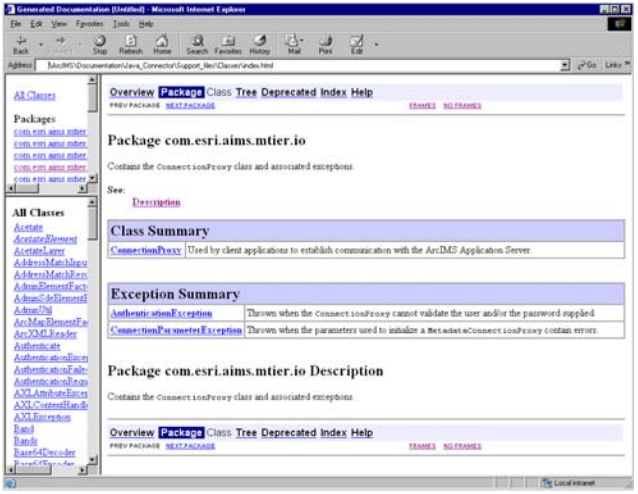
The Overview page is the front page of this API document and provides a list of all packages with a summary for each.



Package

Each package has a page that contains a list of its classes and interfaces, with a summary for each. This page can contain four categories:

- Interfaces (italic)
- Classes
- Exceptions
- Errors



This particular package's page doesn't include a summary of interfaces or errors since none are associated with it.

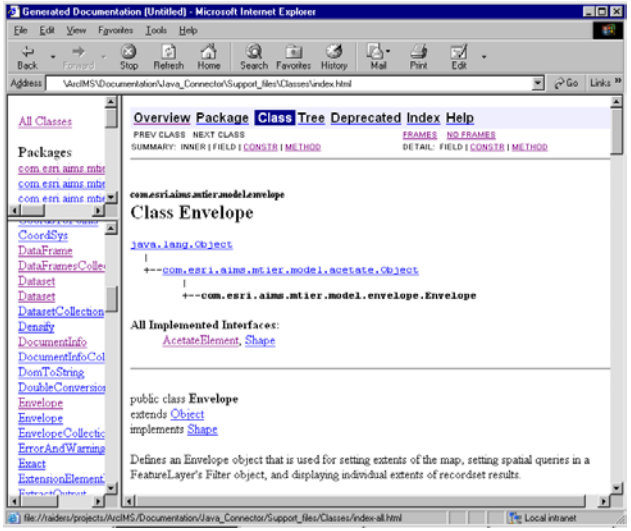
Class/Interface

Every class, interface, and exception in the object model has its own separate page. Each of these pages has three sections consisting of a class/interface/exception description, summary tables, and detailed member descriptions.

Section 1

The top portion of the page gives a general overview of the class, interface, or exception including:

- Class inheritance diagram
- Direct subclasses
- All known subinterfaces
- All known implementing classes
- Class/Interface/Exception declaration
- Class/Interface/Exception description

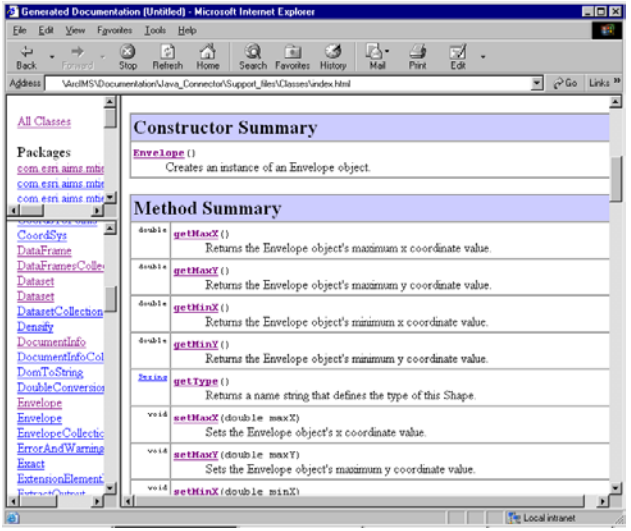


Section 2

The middle section of the page provides a table view of the inner classes, fields, constructors, and methods associated with that particular class, interface, or exception.

- Inner Class Summary
- Field Summary
- Constructor Summary
- Method Summary

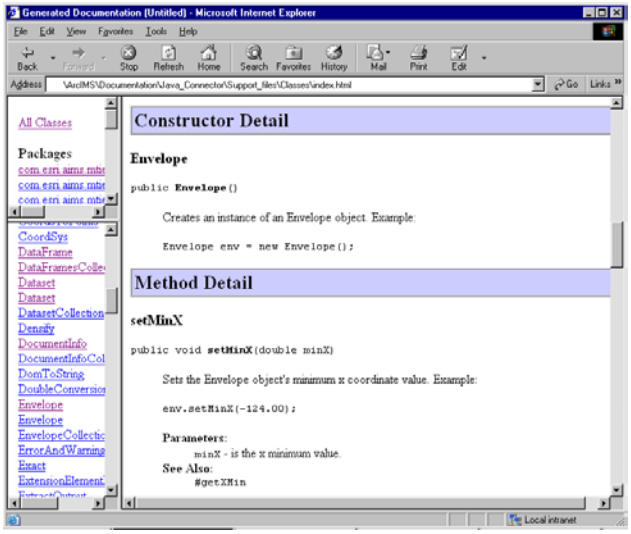
Each summary entry contains the first sentence from the detailed description for that item. The summary entries are alphabetical, while the detailed descriptions are in the order they appear in the source code.



Section 3

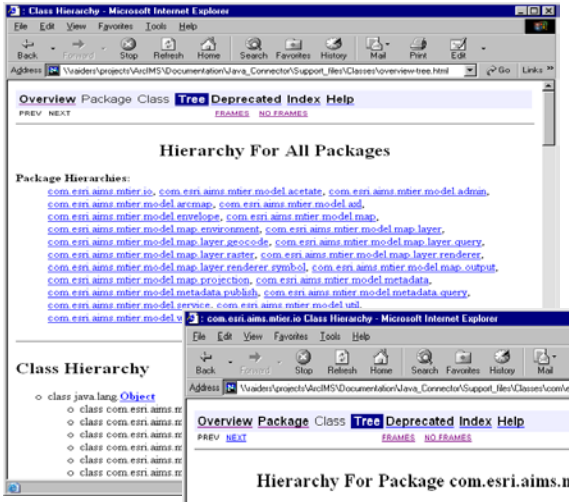
The final section of the page provides detailed descriptions of the fields, constructors, and methods associated with that particular class, interface, or exception. While the summary views in Section 2 are listed in alphabetical order, the detailed descriptions are in the order they appear in the source code.

- Field Detail
- Constructor Detail
- Method Detail



Tree (Class Hierarchy)

There is a Class Hierarchy page for all packages plus a hierarchy for each package. Each hierarchy page contains a list of classes and a list of interfaces. The classes are organized by inheritance structure starting with `java.lang.Object`. The interfaces do not inherit from `java.lang.Object`.

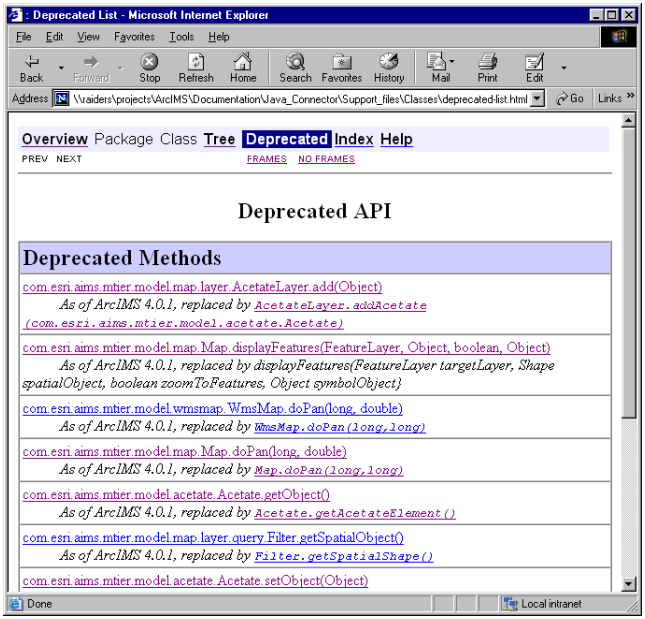


When viewing the Overview page, clicking Tree displays the hierarchy for all packages.

When viewing a particular package, class, or interface page, clicking Tree displays the hierarchy for only that package.

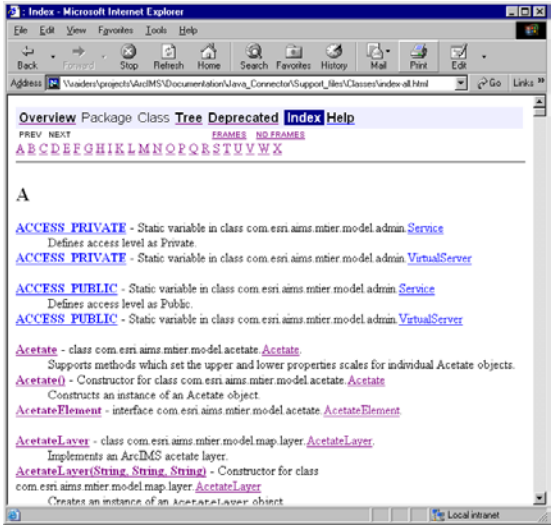
Deprecated API

The Deprecated API page lists all of the APIs that have been deprecated. A deprecated API is not recommended for use, generally due to improvements, and a replacement API is usually given. Deprecated APIs may be removed in future implementations.



Index

The Index contains an alphabetic list of all classes, interfaces, constructors, methods, and fields.



Prev/Next

These links take you to the next or previous class, interface, package, or related page.

Frames/No Frames

These links show and hide the HTML frames. All pages are available with or without frames.

Supported development environments

The Java Connector Object Model beans can be used with many of the IDEs available for Java programmers including Borland's JBuilder, Forte for Java, and Oracle's JDeveloper. Keep in mind, however, that if you are using an IDE, the import and use of the Java Connector Object Model beans can be IDE dependent. Refer to the documentation provided by your chosen IDE for more information.

If you choose not to use an IDE but develop and compile your applications using the command prompt, you need to ensure that the references to arcims_jconnect.jar—the JAR file for the Java Connector Object Model and other dependent JAR files (jsse.jar, jnet.jar, and jcert.jar provided by Sun Microsystems)—are correct.

Object model overview

The Java Connector Object Model beans are easy to use and customize. Detailed information on the methods and classes in the object model is available in the javadoc-generated HTML reference, *ArcIMS—Java Connector Object Model API Specification*. However, an overview of the model is included here.

The Java Connector Object Model supports several packages. Each package, in turn, supports several beans with similar or related functionality as described below:

com.esri.aims.mtier.io

This package contains the ConnectionProxy object and associated exceptions.

A *ConnectionProxy* object is the gateway to using the Java Connector Object Model. This object and its methods are used by client applications to establish communication with an ArcIMS Application Server. The client can choose to use an HTTP, HTTPS, or TCP/IP connection to communicate with an ArcIMS server.

com.esri.aims.mtier.model.auth

This package supports an *Authenticate* object, which is used to provide restricted access to services. This process is called authentication. On initialization, this object reads the Authenticate.properties file and determines if the property *authenticate* is set to True. If true, all client requests have to be associated with a username and password. The username and password will be validated with the ACL, which can be file or JDBC™ based. See 'Enabling Authentication using the Java Connector' in the ArcIMS online Help for more information.

com.esri.aims.mtier.model.acetate

This package contains ArcIMS acetate objects that are used for rendering acetate symbology on a map. You can use the objects defined in this package to:

- Create map layout features, such as North arrow, scalebar, and text objects.
- Create and display geometric shapes, such as line, point, polygon, hole, and ring.
- Perform spatial queries using the geometric shape objects created.

com.esri.aims.mtier.model.admin

This package supports several objects that implement an ArcIMS or ArcSDE service and administers ArcSDE connections.

An *ArcSdeConnection* object implements an object used to store ArcSDE connections.

An *SdeServer* object registers an ArcSDE service.

An *SdeServerCollection* object implements a collection of ArcSDE servers that are administered by an ArcIMS Monitor.

An *SdeStatus* object represents the status of an ArcSDE instance.

A *Server* object implements a proxy object for a site monitor.

A *ServerCollection* represents a collection of server objects.

A *Service* object defines an ArcIMS service.

A *ServiceCollection* object represents a collection of ArcIMS service objects.

A *Site* object represents the access point to a site and all operations on that site.

A *VirtualServer* object implements a proxy for a site's virtual server.

A *VirtualServerCollection* defines a collection of virtual servers.

com.esri.aims.mtier.model.axl

This package contains objects for handling ArcXML exceptions and other related utilities.

com.esri.aims.mtier.model.envelope

This package defines an *Envelope* object, which is used for setting extents of the map, setting spatial queries in a FeatureLayer's Filter object, and displaying the extents of individual recordset results.

com.esri.aims.mtier.model.map

This package defines objects that represent the components of an ArcIMS service. These objects are used extensively for map creation, display, and interaction.

A *Layers* collection defines a collection of layers associated with a service.

A *Legend* object represents a map's legend and supports several methods to manipulate legend-related properties.

A *Map* object represents the map to be displayed. This object supports methods for zooming, panning, setting transparency, and modifying background color among many other functions.

com.esri.aims.mtier.model.map.environment

The objects in this package contain locale information for ArcIMS services. This package includes an *ImageLimit*, *Locale*, *Separators*, and *UIFont* object.

com.esri.aims.mtier.model.map.layer

This package contains objects that represent different ArcIMS layer types.

An *AcetateLayer* object represents an ArcIMS acetate layer.

A *FeatureLayer* object represents an ArcIMS feature layer.

An *ImageLayer* object represents an ArcIMS image layer.

A *Layer* object represents an abstract ArcIMS layer.

com.esri.aims.mtier.model.map.layer.geocode

This package contains objects that are used to perform geocoding operations.

An *AddressMatchInputs* object adds address values to process a geocode request.

An *AddressMatchResults* object represents a collection of geocoded results objects.

A *Result* object represents an individual result. This object is created after a geocode process and contains the score, value, and PointObject containing the resulting coordinates.

com.esri.aims.mtier.model.map.layer.query

This package supports objects that are used to perform queries, for aggregating values based on spatial queries, setting the geometry of features, and creating shapes to perform spatial selects.

A *Buffer* object is used to perform buffers against an ArcIMS service. This object defines several properties that are associated with a buffer.

A *Filter* object supports several methods and properties that are required to perform queries, spatial queries, and buffers against a specific ArcIMS service.

A *Geometry* object supports a collection of points associated with a given recordset. This object also contains methods for creating Line, Polygon, and Ring objects.

A *Records* object contains a collection of recordset values. An instance of Records is created after a filter has been applied to a FeatureLayer object.

A *Recordset* object contains a collection of Records, Geometry, and Envelope objects associated with a specific query against a layer. This object is created after a filter has been applied to a FeatureLayer object.

A *Tabledesc* object is created if the loadRecordset argument in the Map.initMap method is set to True. This object supports several methods to get and set field names, field lengths, field type, and field precision properties.

com.esri.aims.mtier.model.map.layer.renderer

This package implements a renderer object for each ArcIMS ArcXML defined renderer. The renderers included in this package are *Exact*, *GroupRenderer*, *Range*, *Other*, *SimpleRenderer*, *ScaleDependentRenderer*, *SimpleLabelRenderer*, *ValueMap*, *ValueMapRenderer*, and *ValueMapLabelRenderer*.

com.esri.aims.mtier.model.map.layer.renderer.symbol

The objects in this package represent ArcIMS symbols. These symbols help enhance the cartographic appeal of maps and provide the ability to display more information within the limited area of a map. The symbols included in this package are *CalloutMarkerSymbol*, *ChartSymbol*, *HashLineSymbol*, *GradientFillSymbol*, and *ShieldSymbol* among several others.

com.esri.aims.mtier.model.map.output

This package supports individual output objects that specify and contain parameters that describe a response from an ArcIMS service.

An *ExtractOutput* object supports parameters that describe an extract response from an ArcIMS service.

A *LayoutOutput* object is used in conjunction with the ArcIMS ArcMap Server. This object holds the output locations for the associated layout file. The LayoutOutput object extends the Output object.

A *LegendOutput* object is used to set specific parameters when creating a legend image. It is also used for storing the location of the newly created legend image. The LegendOutput object extends the Output object.

A *MapOutput* object contains the output properties of a map image. It also holds the locations for generated map images. The MapOutput object extends the Output object.

An *Output* object represents an abstract output object.

com.esri.aims.mtier.model.map.projection

This package contains objects that are used to reproject the views of an ArcIMS service.

A *CoordSys* object defines the projection coordinate system of a layer.

A *Densify* object sets the interval used for adding points to a layer. The process of data densification adds points to a layer before the layer is projected.

A *FeatureCoordSys* object represents the projection coordinate system that layers in a service or a specific service are projected to.

A *FilterCoordSys* object represents the current coordinate system for layers in a service or a specific service.

com.esri.aims.mtier.model.metadata

This package contains objects that define metadata and implement a metadata document and its collection.

A *Dataset* object represents a metadata document being served.

A *DatasetCollection* represents a collection of metadata documents.

An *Envelope* object implements a wrapper class for a map envelope.

A *GetProp* object reads data from the properties file.

A *User* object defines a metadata user and supplies actions for manipulating users.

com.esri.aims.mtier.model.metadata.publish

Supports objects that are used to publish data to a metadata server.

com.esri.aims.mtier.model.metadata.query

This package supports objects that are used to query a metadata server.

com.esri.aims.mtier.model.util

This object contains utility objects for performing tasks associated with the Java Connector.

A *Base64Encoder* object defines a data encoder.

A *CoordsToPoints* object converts a string of coordinates to a Java Connector Points object.

A *DomToString* object converts an XML document into a String object.

A *DoubleConversion* object provides simple utilities to convert between double and string.

A *FileUpLoad* object is a utility that performs file upload with multipart/form data.

An *InputStreamToString* object converts an InputStream object to a String object.

A *ToStringComparator* object compares two objects using their toString() methods.

com.esri.aims.mtier.model.wmsmap

This package supports objects that are associated with the Open GIS Consortium (OGC) Web Map Service (WMS) 1.1.0 Implementation Specification.

com.esri.aims.mtier.model.workspace

This package contains objects representing the different ArcIMS workspaces available. These objects allow you to add dynamic layers to your services from different data sources.

A *Dataset* object specifies a file to be added as a dynamic layer.

An *ImageWorkspace* object specifies a workspace for image files.

An *SDEWorkspace* object defines an ArcSDE data source.

A *ShapeWorkspace* object defines a shapefile data source.

Java Connector Tag Library

4

IN THIS CHAPTER

- **What's new in the JSP Tag Library for ArcIMS 9.0**
- **What's new in the Java Connector Tag Library for ArcIMS 4.0.1**
- **Java Connector Tag Listing**
- **Java Connector Tag Library conventions**

The ArcIMS—Java Connector Tag Library is a collection of custom JSP tags built on the Java Connector Object Model. These tags provide high-level access to the entire server-side Object Model library. Web developers with little or no knowledge of Java can now develop advanced mapping applications or integrate mapping capabilities into their existing applications using the Tag Library.

The Metadata Explorer is a Web application that demonstrates the capabilities of the Java Connector Tag Library. This browser-based application gives clients the ability to browse and search for geospatial metadata offered in ArcIMS Metadata Services. For more details on the Metadata Explorer, see *Creating and Using Metadata Services*.

An alphabetical listing of JSP tags in the Java Connector Tag Library is included in this chapter.

What's new in the Java Connector Tag Library for ArcIMS 9.0

The following changes have been made to the Java Connector Tag Library for ArcIMS 9.0.

Tags that support the Metadata Server

The following tag has been added to support the Metadata Server. For more information on using the Metadata Server, see *Creating and Using Metadata Services*.

- aims:renameMetadata

Deprecated metadata tags and attributes

The following metadata tags and attributes are no longer valid with Metadata Services and have been deprecated. For backward compatibility, the Spatial Server will process these tags and attributes. However, you will likely receive an incorrect or empty response.

- putMetadataSemantic
  - Element and all attributes no longer supported.
- semanticPair
  - Element and all attributes no longer supported.
- textQuery
  - *zcode* attribute no longer supported.
- valueQuery
  - *zcode* attribute no longer supported.

****

HTML code

<H2>Deprecated metadata tags and attributes</H2>
The following metadata tags and attributes are no longer valid with Metadata Services and have been deprecated. For backward compatibility, the Spatial Server will process these tags and attributes. However, you will likely receive an incorrect or empty response.

<UL>
<LI>putMetadataSemantic

<LI>Element and all attributes no longer supported.
</UL>
<LI>semanticPair

<LI>Element and all attributes no longer supported.
</UL>
<LI>textQuery

<LI><i>zcode</I> attribute no longer supported.
</UL>
<LI>valueQuery

<LI><i>zcode</I> attribute no longer supported.
</UL>
</UL>

Other items of note

- aims:coordSys, aims:featureCoordSys, aims:filterCoordSys - When using definition strings, the quotes in the string must be preceded by a backslash (\) so both the Java Connector and the ArcIMS Spatial Server can interpret the string correctly. For example, the definition string for World Mollweide is:

  PROJCS["World_Mollweide",GEOGCS["GCS_WGS_1984",DATUM["D_WGS_1984",SPHEROID["WGS_1984",6378137,298.257223563]],PRIMEM["Greenwich",0],UNIT["Degree",0.0174532925199433295]],PROJECTION["Mollweide"],PARAMETER["False_Easting",0],PARAMETER["False_Northing",0],PARAMETER["Central_Meridian",0],UNIT["Meter",1]]

  Once the string is modified, it should look like this:

  PROJCS[\"World_Mollweide\",GEOGCS[\"GCS_WGS_1984\",DATUM[\"D_WGS_1984\",SPHEROID[\"WGS_1984\",6378137,298.257223563]],PRIMEM[\"Greenwich\",0],UNIT[\"Degree\",0.0174532925199433295]],PROJECTION[\"Mollweide\"],PARAMETER[\"False_Easting\",0],PARAMETER[\"False_Northing\",0],PARAMETER[\"Central_Meridian\",0],UNIT[\"Meter\",1]]

  Note that the syntax for escaping quotes with the Java Connector is different than that of the Servlet Connector.

What's New in the Java Connector Tag Library for ArcIMS 4.0.1

The Java Connector Tag Library has several new, changed, or deprecated tags and attributes.

Tags that support the Metadata Server

The following tags have been added to support the Metadata Server. For more information on using the Metadata Server, see *Creating and Using Metadata Services*.

- aims:changeMetadataAccess
- aims:deleteMetadataRelationship
- aims:getContentInfo
- aims:changeOwner

The following tags have new attributes.

- aims:documentInfo
  - *content*
- aims:browse
  - *gndextent*
  - *fullOutput*
- aims:search
  - *gndextent*
  - *fullOutput*
- aims:valueQuery
  - *notEqualTo*

The following tags have had changes to attribute values.

- aims:browse
  - *foldermask* values are now 1–7
- aims:createDocument
  - *content* now includes mapFiles and geographicActivies as values
- aims:search
  - *foldermask* values are now 1–7

- aims:searchEnvelope
    - *spatialoperator* values are now within, overlaps, overlaps2, fuzzywithin, and fuzzyequals

New tags and attributes for administration

The following tags have been added to support administration.

- aims:getStartupParameter

The following tags have new attributes.

- aims:getContainerAttribute
    - *virtualServerType*
    - *virtualServerVersion*
- aims:createVirtualServer
    - *recyclingFrequency*
    - *recyclingHour*
    - *recyclingMinute*

New and changed tags to support ArcMap layouts

The following tags have been added to support the ArcMap layouts.

- aims:dataframe
- aims:getArcMapLayout
- aims:getArcMapLayoutAttribute
- aims:layerList
- aims:layoutOutput
- aims:scale

The following tags have new attributes.

- aims:iterateDataframe
    - *dataframeCount*

The following tags and attributes have been deprecated in ArcIMS 4.0.1.

- aims:arcMapService
    - *outputType* deprecated

    - *loadLayout* deprecated
- aims:getLayout deprecated

New tags to support image streaming (Image Server only)

The following attributes have been added to support image streaming.

- aims:map
    - *stream*
    - *streamType*
    - *legendStream*
    - *legendStreamType*

New tags for map display

The following new tag supports map display.

- aims:iterateMapService
- aims:getEnvelope

Tags with new attributes

The following tags have new attributes.

- aims:layer
    - *extractable*
- aims:map
    - *autoresize*
    - *drawmap*
    - *dpi*
- aims:displayFeatures
    - *whereExpression*
    - *filter*
- aims:iterateLayers
    - *reverse*
- aims:getLayerAttribute
    - added new value FEATURETYPE to *attribute*

Tags with new nested tags

- aims:buffer
    - aims:gradientFillSymbol, aims:hashlineSymbol, aims:rasterFillSymbol, aims:rasterMarkerSymbol, simpleLineSymbol, aims:simpleMarkerSymbol, aims:simplePolygonSymbol, aims:truetypeMarkerSymbol
- aims:displayFeatures
    - envelope
- aims:targetLayer
    - aims:gradientFillSymbol, aims:hashlineSymbol, aims:rasterFillSymbol, aims:rasterMarkerSymbol, simpleLineSymbol, aims:simpleMarkerSymbol, aims:simplePolygonSymbol, aims:truetypeMarkerSymbol

Tags with nested tags removed

- aims:bufferLayer now has only aims:simplePolygonSymbol as a nested tag

Tags with attributes removed

The following tags had the *file* attribute removed.

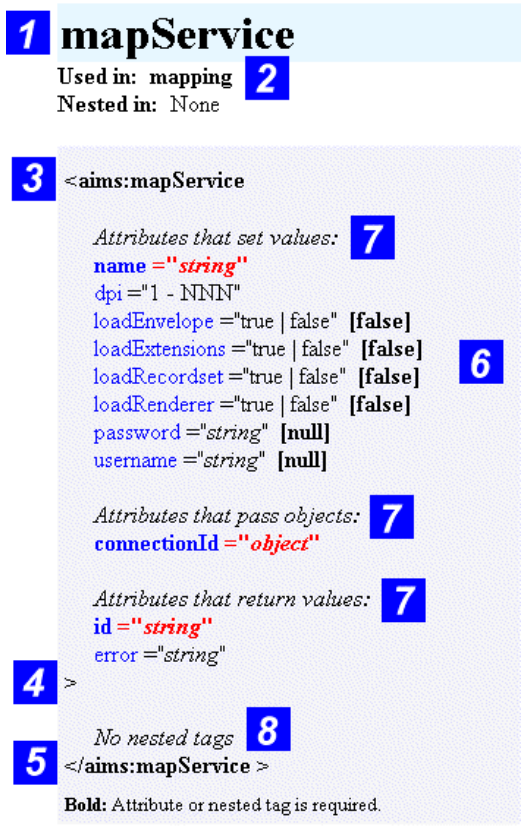
- aims:extractOutput
- aims:legendOutput
- aims:mapOutput

Java Connector Tag listing

Java Connector Tag Library conventions

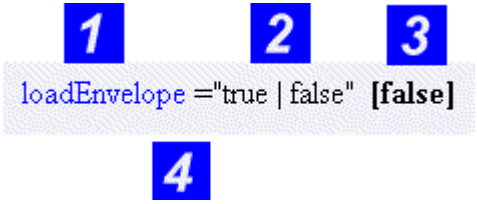
The Java Connector Tag structure includes the tag name, attributes, and any nested tags.

Tag name



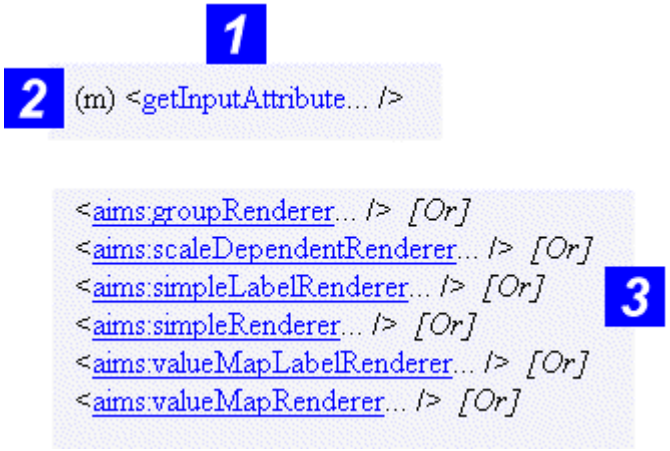
1. The name of the tag.
2. The group to which the tag belongs. The JSP Tag Library is divided into three general groups: mapping, administration, and metadata.
3. The opening tag starts with a less than symbol (<) and is followed by its name. Note that all tag names must be preceded by a prefix. The prefix used in all the examples is "aims:". Tag names are case sensitive.
4. The opening tag ends with a greater than symbol (>).
5. The closing tag is similar to the opening element. It begins with the less than symbol and a forward slash (/) and is followed by the prefix and its name. The tag ends with a greater than symbol.
6. If the tag includes any attributes, they are included after the opening tag.
7. Attributes are divided into three types: attributes that set values, attributes that return values, and attributes that pass objects.
8. All nested tags are inserted before the closing element.

Attributes



1. Attributes are case sensitive. If the attribute is required, it is bold. A definition of each attribute is found in the attribute table for the tag.
2. If an attribute has a defined list of valid values, all possible values are listed. Only one value can be used at a time. Attribute values are always placed within double quotes (").
3. If an attribute has a default value, it is listed in brackets after the attribute list or type.
4. If an attribute does not have a list of known values, the value type is listed, such as double, integer, and string.

Nested tags



1. Nested tags are case sensitive. If a nested tag is required, it is bold. Each nested tag is linked to the corresponding page describing that tag.
2. Some nested tags can be used multiple times. If this is the case, the letter "m" in parentheses (m) is in front of the tag.
3. In some cases, special instructions are given in brackets after the element. The most common scenario is when a group of nested tags is listed, but only one tag can be used. In this example, [Or] means to select one tag from the group. More details are given in the Restrictions section to explain special instructions.

acetateObject

Used in: mapping
Parent elements: aims:createLayer

```
<aims:acetateObject

    Attributes that set values:

    When using ArcMap Server:
    units ="database | pixel"  [database]

    When using Image Server:
    lower ="string"  [1:1]
    units ="database | pixel"  [database]
    upper ="string"  [1:infinity]
>

    <coordSys... />
    <line... />
    <northArrow... />
    <point... />
    <polygon... />
    <scalebar... />
    <text... />

</aims:acetateObject >
```

Description:

Defines the scale thresholds and units of an object to be placed on an acetate layer.

Restrictions:

None

Notes:

None

Attribute descriptions for acetateObject:

Attribute	Usage
lower	Minimum scale to display acetateObject using a relative scale, such as 1:24,000. Scale can also be calculated as the number of map units per pixel.
units	Determines how coordinates for the object are specified. Coordinates can be specified two ways: • Database. Refers to positioning an object using x,y coordinates in the coordinate system of the ArcIMS service or request. For example, if the service is in Robinson, then the coordinates for the object should also be in Robinson. If the coordinates for the object are different from the coordinate system used in the service or request, then aims:coordSys should be used. <aims:acetateObject...> <aims:coordSys.../> ... </aims:acetateObject> • Pixel. Refers to positioning an object using x,y coordinates in pixels. The pixels along the left edge of the map frame have an x-coordinate of zero. The pixels along the bottom edge have a y-coordinate of zero. /UL>
upper	Maximum scale to display Acetate Object using a relative scale, such as 1:24,000. Scale can also be calculated as the number of map units per pixel.

Examples for acetateObject:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300"/>
<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="serviceName"
/>

<aims:map id="myMap" serviceId="<%=myMapService%>" width="300" height="200"
background="100,140,230" transcolor="0,0,0">
```

```
  <aims:createLayer layerId="LayerNorthArrow" name="AcetateLayer" type="acetate"
visible="true">
    <aims:acetateObject units="pixel">
      <aims:northArrow x="50" y="50" type="3" shadow="230,230,230" />
    </aims:acetateObject>
    <aims:acetateObject units="pixel">
      <aims:line coords="50 10;250 10" >
        <aims:hashLineSymbol color="0,0,0" lineThickness="3" tickThickness="3"
transparency="0.5" interval="16" width="16" type="foreground" antialiasing="false"
overlap="true" />
      </aims:line>
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>
<img src="<%=(myMap)%>" />
```

addRelevance

Used in: metadata
Parent elements: None

```
<aims:addRelevance
    Attributes that set values:
    docId ="string"
    relevance ="1 - 10"

    Attributes that pass objects:
    connectionId ="object"
    service ="object"
>

    No Child Elements
</aims:addRelevance >
```

Bold: Attribute or child element is required.

Description:

Sends a request to the server to increment a particular document's relevance.

Restrictions:

None

Notes:

None

Attribute descriptions for addRelevance:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
docId	String used to uniquely identify a document. The client used to publish the metadata is responsible for creating the document ID. This ID is automatically assigned when using ArcCatalog™. If another client is used, aims:getUUID can be used to request a valid ID. The format for an ID is the following: {HHHHHHHH-HHHH-HHHH-HHHH-HHHHHHHHHHHH} where H is a hexadecimal digit (0–9,a–f,A–F). The ID is limited to 38 characters.
relevance	Value between 1 and 10 indicating the relative relevance of the document to the client. The value is added to the existing value for a new relevance total.
service	The name of the metadata service.

Examples for addRelevance:

Example 1: Send a request to the server to increment a particular document's relevance.

```
<aims:getSettings id="hostName" value="meta_host_name" />
<aims:getSettings id="portNumber" value="meta_port_number" />
<aims:getSettings id="serviceName" value="meta_service_name" />

<aims:tcpConnection id="myConnection" host="<%= hostName %>" port="<%= portNumber %>"
debug="true" />

<% String docId = request.getParameter("docId"); %>

<aims:addRelevance relevance="4" connectionId="<%= myConnection %>" service="<%=
serviceName %>" docId="<%= docId %>" />
```

addressMatchInputs

Used in: mapping
Parent elements: aims:layer

```
<aims:addressMatchInputs

    Attributes that set values:
    maxCandidates ="integer"  [20]
    minScore ="0 – 100"  [60]

    Attributes that return values:
    id ="string"
>

    (m) <setInput... />

</aims:addressMatchInputs >
```

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Reads in the address match inputs from the GET_GEOCODE request and sets a value for the matching ID.

Restrictions:

AddressMatchInputs must be nested in a layer that contains a geocoding index.

Notes:

LoadExtensions must be set to "true" in aims:mapService to read the inputs.

Attribute descriptions for addressMatchInputs:

Attribute	Usage
id	Returns the AddressMatchInputs object.
maxCandidates	Maximum number of returned candidates.
minScore	Minimum score of returned candidates. If not included, all candidates with scores above 60 are returned. A candidate with a score of 100 means a perfect match, and 0 means no match.

Examples for addressMatchInputs:

Example 1:

```
<aims:tcpConnection id="mytcpConnection" port="5300" host="brugge" debug="true" />

<aims:mapService id="myService" connectionId="<%=mytcpConnection%>" name="geostreets"
loadExtensions="true" />

<aims:map id="mapImage" serviceId="<%=myService%>" reset="false" >
  <aims:layer layerId="0" >
    <aims:addressMatchInputs id="myInputs" minScore="0" maxCandidates="100" >
      <aims:setInput inputId="STREET" value="380 New York St" />
      <aims:setInput inputId="ZONE" value="" />
      <aims:setInput inputId="CROSSSTREET" value="" />
    </aims:addressMatchInputs>
        <table>
    <aims:iterateAddressMatchInputs inputs="<%=myInputs%>" count="myCount"
minScore="myMinScore" maxCandidates="myMax" style="myStyle" >
      <aims:getInputAttribute id="myDes" attribute="DESCRIPTION" />
      <aims:getInputAttribute id="myId" attribute="ID" />
      <aims:getInputAttribute id="myLabel" attribute="LABEL" />
      <aims:getInputAttribute id="myType" attribute="TYPE" />
      <aims:getInputAttribute id="myWidth" attribute="WIDTH" />
            <tr><td>
        <b>desc is: <%=myDes%></b><br>
           id is: <%=myId%><br>
```

```
            label is: <%=myLabel%><br>
            type is: <%=myType%><br>
            width is: <%=myWidth%>
             </td></tr>
    </aims:iterateAddressMatchInputs>

       <tr><td>
      <b>INPUT count is: <%=myCount%></b><br>
         minScore is: <%=myMinScore%><br>
         maxCandidates is: <%=myMax%><br>
         style is: <%=myStyle%>
    </td></tr>

    <aims:iterateAddressMatchResults inputs="<%=myInputs%>" count="resultsCount" >
      <aims:getResultAttribute id="myValue" attribute="VALUE" />
      <aims:getResultAttribute id="myScore" attribute="SCORE" />
      <aims:getResultAttribute id="myPoint" attribute="POINT" />
      <tr><td>
        <b>value is: <%=myValue%></b><br>
           score is: <%=myScore%><br>
           point is: <%=myPoint%>
      </td></tr>
    </aims:iterateAddressMatchResults>

        <tr><td>
      <br><b>RESULT count is: <%=resultsCount%></b>
        </td></tr></table>
  </aims:layer>
</aims:map>
```

adminServer

Used in: administration
Parent elements: None

```
<aims:adminServer

    Attributes that set values:
    action ="addContainer | removeContainer"
    name ="string"
    removeId ="string"

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    error ="string"
    addedId ="string"
>

    No Child Elements
</aims:adminServer >
```

Bold: Attribute or child element is required.

Description:

Administers the ArcIMS Spatial Servers for a server machine.

Restrictions:

None

Notes:

None

Attribute descriptions for adminServer:

Attribute	Usage
action	Defines whether to add or remove an ArcIMS Spatial Server.
addedId	Returns a string identifying the new ArcIMS Spatial Server name.
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
error	Returns an error string if any occurred. Returns "Success" if no error.
name	The name of the server machine to administer.
removeId	The name of the ArcIMS Spatial Server to remove.

Examples for adminServer:

Example 1: Add and remove an ArcIMS Spatial Server.

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://mymachine"
      role="admin"
      username="admin"
      password="admin"
/>

<aims:adminServer name="mymachine" action="addContainer" addedId="myId"
connectionId="<%=myhttpConnection%>" error="addError" />

<%=myId%>

<aims:adminServer name="mymachine" action="removeContainer" removeId="<%=myId%>"
connectionId="<%=myhttpConnection%>" error="removeError" />
```

adminService

Used in: administration
Parent elements: None

```
<aims:adminService

    Attributes that set values:
    action ="start | stop | remove"
    service ="string"

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    error ="string"
>

    No Child Elements
</aims:adminService >
```

Bold: Attribute or child element is required.

Description:

Administers a service from a site.

Restrictions:

None

Notes:

The *service* attribute can be retrieved from aims:getService, getServices, or createService. A Service or ServiceCollection object is returned.

Attribute descriptions for adminService:

Attribute	Usage
action	Defines whether to start, stop, or remove the service.
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
error	Returns an error string if any occurred. Returns "Success" if no error.
service	The name of service to administer.

Examples for adminService:

Example 1: Stop, start, or remove a service.

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://brugge"
      role="admin"
      username="admin"
      password="admin"
/>

<aims:getService id="myService" connectionId="<%=myhttpConnection%>" name="World"
error="error1"/>

<aims:adminService service="<%=myService%>" connectionId="<=myhttpConnection%>"
action="stop" error="error2" />

<aims:adminService service="<%=myService%>" connectionId="<=myhttpConnection%>"
action="start" error="error3" />

<aims:adminService service="<%=myService%>" connectionId="<=myhttpConnection%>"
action="remove" error="error4" />
```

arcMapService

Used in: **mapping**
Parent elements: None

```
<aims:arcMapService

    Attributes that set values:
    name ="string"
    dpi ="1 - NNN"
    loadEnvelope ="true | false"  [false]
    loadExtensions ="not used"
    loadLayout ="true | false"
    loadRecordset ="true | false"  [false]
    loadRenderer ="not used"
    outputType ="ai | bmp | emf | eps | jpg | pdf | png8 | png24"
    password ="string"
    username ="string"

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    id ="string"
    error ="string"
>

    <envelope... />

</aims:arcMapService >
```

Bold: Attribute or child element is required.

Description:

Reads initial values from an ArcMap Image Service such as envelopes, layout information, and database field names.

Restrictions:

The nested tag aims:envelope has been deprecated in ArcIMS 4.0.1. Use aims:envelope as a nested tag to aims:getArcMapLayout instead.

Notes:

- When using aims:envelope as a nested tag, the units are in page units rather than map units. This envelope is referencing an ArcMap layout rather than a map.
- For reading initial values from an Image Service, see aims:mapService.

Attribute descriptions for arcMapService:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
dpi	Dots per inch. Used for calculating the correct scale thresholds for scale-dependent elements. The *dpi* value overrides the value used in a service.
error	Returns an error string if any occurred. Returns "Success" if no error.
id	Returns a Data Frames Collection object.
loadEnvelope	Loads individual layer envelopes from service.
loadExtensions	Not used in this release of ArcIMS.
loadLayout	Loads the layout information. This attribute has been deprecated in ArcIMS 4.0.1. Use aims:getArcMapLayout instead.
loadRecordset	Loads individual layer recordsets from service.
loadRenderer	Not used in this release of ArcIMS.
name	ArcIMS service name.
outputType	Output file type. Can be ai, bmp, emf, eps, jpg, pdf, png8 (8 bit), png24 (24 bit). This attribute has been deprecated in ArcIMS 4.0.1. Use aims:layoutOutput instead.
password	Password if authentication required.
username	Username if authentication required.

Examples for arcMapService:

Example 1:

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>arcMapService Sample</title>
</head>
<body>
<aims:tcpConnection id="myConnection" host="localhost" port="5300" debug="true" />

<aims:arcMapService id="arcmyDataframeCollection" connectionId="<%=(myConnection)%>"
name="arcMapService" error="arcmyError" />

<aims:iterateDataframe id="myMapObject" dataframeName="dfn" dataframeCount="dfc"
dataframeCollection="<%=(arcmyDataframeCollection)%>">
  Dataframe Name : <%=(dfn)%><br>
  Dataframe Count : <%=(dfc)%><br>
  <aims:map id="myMap" serviceId="<%=(myMapObject)%>" width="300" height="100" />
  <img src="<%=myMap%>" /><br>
</aims:iterateDataframe>

</body>
</html>
```

browse

Used in: metadata
Parent elements: None

<aims:browse

Attributes that set values:
batchSize ="*integer*"
docId ="*string*"
folderMask ="1 - 7" **[7]**
fullOutput ="true | false" **[true]**
gndextent ="none | document | search" **[none]**
sort ="name | relevance | contenttype | local_area | global_area" **[name]**
sort2 ="name | relevance | contenttype | local_area | global_area"
startBatchAt ="0 - NNN" **[0]**
startResult ="*integer*" **[0]**
type ="children | parents | siblings | ancestors | descendants" **[children]**

Attributes that pass objects:
connectionId ="*object*"
service ="*string*"

Attributes that return values:
id ="*object*"
numResults ="*integer*"
totalResults ="*integer*"
>

No Child Elements
</aims:browse >

Bold: Attribute or child element is required.

Description:

Returns a subset of documents.

Restrictions:

None

Notes:

None

Attribute descriptions for browse:

Attribute	Usage
batchSize	The maximum number of results to return.
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
docId	String used to uniquely identify a document. The client used to publish the metadata is responsible for creating the document ID. This ID is automatically assigned when using ArcCatalog. If another client is used, aims:getUUID can be used to request a valid ID. The format for an ID is the following: {HHHHHHHH-HHHH-HHHH-HHHH-HHHHHHHHHHHH} where H is a hexadecimal digit (0–9,a–f,A–F). The ID is limited to 38 characters.
folderMask	Specifies folder types to be returned in the response. The value of *foldermask* is an integer storing the sum of one or more of the following values: • 1 = Root document • 2 = "Normal" folder • 4 = Document For example, use "4" to return documents only. Use "7" (1 + 2 + 4) to return all types of folders. The different values are: • 1 = Root document only (1) • 2 = "Normal" folders only (2) • 3 = Root document and "normal" folders (1 + 2)

	• 4 = Documents only (4) • 5 = Root document and documents (1 + 4) • 6 = "Normal" folders and documents (2 + 4) • 7 = Root document, "normal" folders, and documents (1 + 2 + 4)
fullOutput	By default this attribute is "true". When "true", an XML file is generated, and a thumbnail and GND file is created if available. When "false", the XML file, thumbnail, and gnd files are not created.
gndextent	The extent written to the GND file. When "none" is selected, the default extent of the service is used. For "document" the extent is taken from the metadata document. For "search" the extent is the search extent specified in the client such as Metadata Explorer.
id	Returns a DataCollection object.
numResults	Returns a value containing the number of results returned in this batch.
service	The name of the metadata service.
sort	Preference for ordering results. "Name" orders the results alphabetically. "Relevance" lists results from highest to lowest relevance. "Contenttype" sorts and groups results by content type. "Localarea" lists results by area in ascending order. "Globalarea" lists results by area in descending order.
sort2	Sorts search results that were batched using startresult and maxresults. "Name" orders the results alphabetically. "Relevance" lists results from highest to lowest relevance. "Contenttype" sorts and groups results by content type. "Localarea" lists results by area in ascending order. "Globalarea" lists results by area in descending order.
startBatchAt	By default, all records meeting the search criteria are returned starting with record 0. This attribute allows a specified record as the start record.
startResult	Returns a value containing the starting index of the batched results.
totalResults	Returns a value containing the total number of results found.
type	The type of documents to browse for.

Examples for browse:

Example 1: Returns a subset of documents.

```
<aims:getSettings id="hostName" value="meta_host_name" />
<aims:getSettings id="portNumber" value="meta_port_number" />
<aims:getSettings id="serviceName" value="meta_service_name" />
```

```
<aims:tcpConnection id="myConnection" host="<%= hostName %>" port="<%= portNumber %>"
debug="true" />

<% String docId = request.getParameter("docId"); %>

<%-- Retrieve a handle to the children of the specified document --%>
<aims:browse
      id="browseChildren"
      connectionId="<%= myConnection %>"
      service="<%= serviceName %>"
      docId="<%= docId %>"
      folderMask="2" />

<%-- Iterate through the list of children, printing out for each their name --%>
<% if (browseChildren.size() > 0) { // print selected folder's children %>

      <aims:iterateMetadata iterate="<%= browseChildren %>" size="mySize">

            <% if (!printedHeader) { %>
                  <span class="explorerText">Categories</span>
                  <ul>
                  <% printedHeader = true; %>
            <% } %>

            <aims:datasetProperty id="theName" value="NAME" />
            <aims:datasetProperty id="theDocId" value="DOCID" />

            <li><a href="JavaScript:browseFolder('<%= URLEncoder.encode(theName) %>','<%=
URLEncoder.encode(theDocId) %>');"><%= theName %></a></li>

      </aims:iterateMetadata>

<% } %>
```

buffer

Used in: mapping
Parent elements: aims:filter

```
<aims:buffer
    Attributes that set values:
    distance ="double"
    bufferUnits ="decimal_degrees | miles | feet | kilometers | meters"
    performBuffer ="true | false"
    project ="true | false"  [true]
>

    <bufferLayer... />
    <targetLayer... />
    <gradientFillSymbol... />  [Or]
    <hashLineSymbol... />  [Or]
    <rasterFillSymbol... />  [Or]
    <rasterMarkerSymbol... />  [Or]
    <simpleLineSymbol... />  [Or]
    <simpleMarkerSymbol... />  [Or]
    <simplePolygonSymbol... />  [Or]
    <truetypeMarkerSymbol... />  [Or]

</aims:buffer >
```

Bold: Attribute or child element is required.

Description:

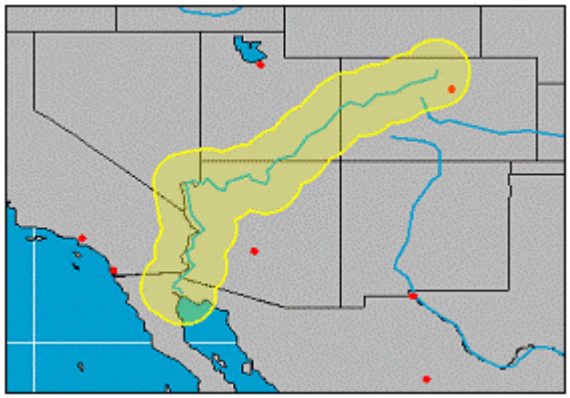
Builds a buffer region around selected features.

Restrictions:

- Aims:filter used in aims:buffer cannot contain another buffer inside.
- **Known limitation with the Java Connector.** When using a target layer, the target layer cannot be the same layer as the buffer layer. This limitation is with the Java Connector and not with ArcXML. Using ArcXML, the target and buffer layers can be the same.
- **Known limitation with the Java Connector Tag Library.** When using *subfields* in aims:filter, the subfields are ignored when using aims:buffer. All fields are returned.

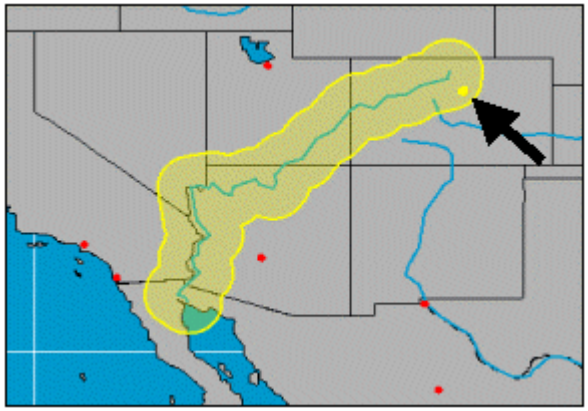
Notes:

- Aims:buffer can be used three ways:



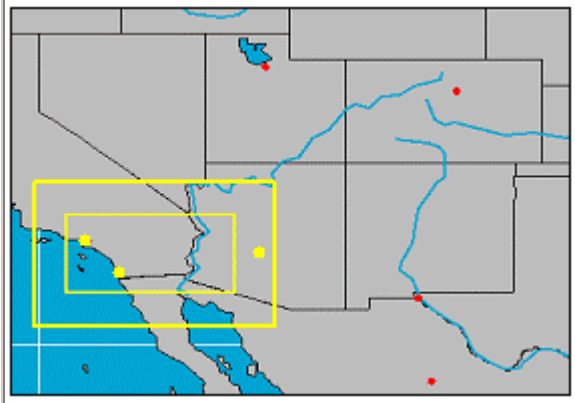
To buffer a region around one or more selected features, use aims:buffer inside aims:filter. In this example, the Colorado River is buffered.

```
<aims:layer visible="true"
layerId="Rivers">
  <aims:filter whereExpression="NAME =
'Colorado'">
    <aims:buffer distance="80"
bufferUnits="MILES" performBuffer="true">
      <aims:bufferLayer>
        <aims:simplePolygonSymbol
fillColor="255,255,0" fillTransparency=".3"
boundaryColor="255,255,0" boundaryWidth="2"
/>
      </aims:bufferLayer>
    </aims:buffer>
  </aims:filter>
</aims:layer>
```



To select features from another layer that fall within the buffer region, use aims:targetLayer as a nested tag. In this example, cities within 80 miles of the Colorado River are selected.

```
<aims:layer visible="true"
layerId="Rivers">
  <aims:filter whereExpression="NAME =
'Colorado'">
     <aims:buffer distance="80"
bufferUnits="MILES" performBuffer="true">
      <aims:bufferLayer>
        <aims:simplePolygonSymbol
fillcolor="255,255,0" filltransparency=".3"
boundarycolor="255,255,0"
boundarywidth="2"/>
      </aims:bufferLayer>
      <aims:targetLayer
targetLayerId="Cities" />
        <aims:simpleMarkerSymbol
color="255,255,0" width="6" />
      </aims:targetLayer>
    </aims:buffer>
  </aims:filter>
</aims:layer>
```



To use a buffer as a selection tool, aims:buffer is used around a defined filter, such as a point or envelope. In this example, aims:buffer is used to buffer around an envelope. The inner yellow box is the original envelope. The outer yellow box is the buffer filter. Note that these boxes would not normally display—only the selected items within the filter.

```
<aims:layer visible="true" layerId="5">
  <aims:filter >
    <aims:buffer distance="3"
bufferUnits="MILES" performBuffer="false"
/>
    <aims:envelope minx="-121.98"
miny="37.265" maxx="-121.823" maxy="37.398"
/> <%--the envelope to buffer --%>
  </aims:filter>
</aims:layer>
```

- When using *subfields* in aims:filter, the subfields apply to the target layer if a target layer is present. If no target layer is present, the subfields apply to the buffer layer. The field #SHAPE# or #ALL# must be included in the subfields list.
- A buffer distance of 0 can be used on polygon features. If a buffer distance of 0 is used with line or point features, no data is returned.
- Buffer results may be inaccurate on unprojected data. It is recommended to project data before applying a buffer.

Attribute descriptions for buffer:

Attribute	Usage
bufferUnits	Specifies units that apply to buffer.
distance	Buffer area width in buffer units.
performBuffer	Performs a buffer operation on the given request.
project	Generated buffers are projected if aims:featureCoordSys is in the request. If *project* is set to "false", the buffer is not projected.

Examples for buffer:

Example 1: When displaying a buffer on a map.

```
<%@page contentType="text/html"%>
<%@taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head><title>buffFeature.jsp</title></head>
<body>

<%--Buffer Calabazas River in SantaClara 2 miles.--%>

<aims:tcpConnection id="mytcpConnection" host="localhost" port="5300" debug="true" />
<aims:mapService id="myMapService" connectionId="<%=mytcpConnection%>" name="SantaClara"
                 loadEnvelope="true" loadExtensions="true" loadRecordset="true"
loadRenderer="true" />
<aims:map id="myMap" serviceId="<%=myMapService%>" width="500" height="350"
            envelope="myExtent" legend="myLegend" bufferImage="true"
            bufferRegion="true" createMap="true">
    <aims:envelope minx="-122.121" miny="37.203" maxx="-121.895" maxy="37.466" />
    <aims:layer visible="true" layerId="3"> <%--ID 3 is rivers layer.--%>
        <aims:filter whereExpression="NAME = 'CALABAZAS CREEK'">
            <aims:buffer distance="2" bufferUnits="MILES" performBuffer="true">
                <aims:bufferLayer>
                    <aims:simplePolygonSymbol fillColor="0,255,0" fillTransparency="0.3"
fillType="SOLID"/>
                </aims:bufferLayer>
                <aims:simpleLineSymbol transparency="1.0" type="dash" width="5"
capType="round" joinType="round" color="0,0,255"  />
            </aims:buffer>
        </aims:filter>
    </aims:layer>
</aims:map>

<img src="<%=myMap%>" >
</body>
</html>
```

Example 2: When displaying a buffer on a map along with selected features in the target layer.

```
<%@page contentType="text/html"%>
<%@taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head><title>Buffer Feature</title></head>
<body>

<%--Buffer Belgium 100 miles and select all the cities within.--%>

<aims:tcpConnection id="mytcpConnection" host="localhost" port="5300" debug="true" />
<aims:mapService id="myMapService" connectionId="<%=mytcpConnection%>" name="SearchMap"
loadEnvelope="true" loadExtensions="true" loadRecordset="true" loadRenderer="true" />

<aims:map id="myMap" serviceId="<%=myMapService%>" width="500" height="350"
envelope="myExtent" legend="myLegend" bufferImage="true" bufferRegion="true"
createMap="true">
  <aims:envelope minx="0.933" miny="47.129" maxx="8.608" maxy="54.764" />
  <aims:layer visible="true" layerId="0"> <%--ID 1 is countries layer.--%>
    <aims:filter relation="area_intersection">
      <aims:buffer distance="100" bufferUnits="MILES" performBuffer="true">
        <aims:bufferLayer>
          <aims:simplePolygonSymbol transparency="0.5" fillColor="0,255,0"
fillType="HORIZONTAL"/>
        </aims:bufferLayer>
        <aims:targetLayer targetLayerId ="2" > <%--ID 2 is cities layer--%>
          <aims:simpleMarkerSymbol color="255,0,0" type="star" width="20" />
        </aims:targetLayer>
      </aims:buffer>
      <aims:envelope minx="4.68" miny="50.34" maxx="5.46" maxy="50.86" />
    </aims:filter>
  </aims:layer>
</aims:map>

<img src="<%=myMap%>" >
</body>
</html>
```

Example 3: When displaying a map and attributes of the target layer.

```
<%@taglib prefix="aims" uri="arcims_taglib.tld" %>
<html>
<head>
<title>ArcIMS JSP Connector - Buffer</title>
</head>
<body>
<%
  int i=0;
  String featureCount = "0";
%>
<aims:tcpConnection id="myConnection" host="localhost" port="5300"/>
<aims:mapService id="myMapService" connectionId="<%=(myConnection)%>" name="SantaClara"
loadEnvelope="true" loadExtensions="true" loadRecordset="true" loadRenderer="true"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" envelope="myEnvelope"
legend="myLegend" width="500" height="350" bufferImage="true" bufferRegion="true" >
<aims:separators ts="," cs=" " />
<aims:layer layerId="3" visible="true"> <!-- River Layer -->
  <aims:filter whereExpression="NAME = 'STEVENS CREEK'"  >
    <aims:buffer distance="5" bufferUnits="Miles" performBuffer="true">
      <aims:bufferLayer>
        <aims:simplePolygonSymbol transparency="0.5" fillType="" fillTransparency="0.5"
fillColor="255,255,0" boundaryColor="0,0,255"/>
      </aims:bufferLayer>
      <aims:targetLayer targetLayerId="5" > <!-- Cities Layer -->
        <aims:simpleMarkerSymbol width="25" color="0,255,255" type="star"/>
      </aims:targetLayer>
   </aims:buffer>
  </aims:filter>
  <aims:recordset>
    <table border="1"><tr>
    <aims:iterateTableDesc fieldName="fName" fieldType="fType" fieldPrecision="fPrec"
fieldLength="fLen">
     <td bgcolor="#ffe6a6"><%=(fName)%></td>
    </aims:iterateTableDesc>
  </aims:recordset>
     </tr><tr>
  <aims:recordset>
```

```
    <aims:iterateRecordset recordsetCount="myCount" recordLength="recLength"
fieldValue="fValue">
<%
    if (i % (Integer.parseInt(recLength))  == 0) {
%>
      </tr><tr><td><%=(fValue)%></td>
<%
    }else{
%>
      <td><%=(fValue)%></td>
<%
    }
    i++;
    featureCount = myCount;
%>
    </aims:iterateRecordset>
  </aims:recordset>
</table><br>
There are <font color="red"><%=featureCount%> </font>cities within 5 miles buffer of
Stevens Creek.
</aims:layer>
</aims:map>
<center><img src="<%=(myMap)%>"/></center>
</body>
</html>
```

bufferLayer

Used in: mapping
Parent elements: aims:buffer

<bufferLayer >
No Attributes

<simplePolygonSymbol... />

</aims:bufferLayer >

Description:

Main tag for setting up buffer layer properties.

Restrictions:

Only one symbol can be used per request.

Notes:

None

Examples for bufferLayer:

Example 1:

```
<aims:tcpConnection id="myConnection" host="localhost" port="5300" debug="true" />
<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="SantaClara"/>

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300"
bufferImage="true" bufferRegion="true">
  <aims:layer layerId="5" visible="true">
    <aims:filter whereExpression="NAME='San Jose'">
      <aims:buffer distance="15" bufferUnits="Miles" performBuffer="true" >
```

```
        <aims:bufferLayer>
          <aims:simplePolygonSymbol fillTransparency="1.0" fillType="HORIZONTAL"
boundaryColor="0,0,0" fillColor="0,0,0"/>
        </aims:bufferLayer>
        <aims:targetLayer targetLayerId="4">
          <aims:simpleMarkerSymbol type="STAR" width="20" color="0,0,255" />
        </aims:targetLayer>
      </aims:buffer>
    </aims:filter>
  </aims:layer>
</aims:map>
<img src="<%=(myMap)%>" />
```

calloutMarkerSymbol

Used in: mapping
Parent elements: aims:exact aims:other aims:range aims:simpleLabelRenderer

```
<aims:calloutMarkerSymbol
    Attributes that set values:
    antialiasing ="true | false"  [false]
    backColor ="0,0,0 - 255,255,255"  [255,255,255]
    boundaryColor ="0,0,0 - 255,255,255"  [0,0,0]
    font ="Any system font"  [Arial]
    fontColor ="0,0,0 - 255,255,255"  [0,0,0]
    fontSize ="1 - NNN"  [12]
    fontStyle ="regular | bold | italic | underline | outline | bolditalic"  [regular]
    glowing ="0,0,0 - 255,255,255"
    interval ="0 - NNN"  [10]
    outline ="0,0,0 - 255,255,255"
    shadow ="0,0,0 - 255,255,255"
    transparency ="0.0 - 1.0"  [1.0]
>

    No Child Elements
</aims:calloutMarkerSymbol >
```

Description:

Creates a callout box around each label.

Restrictions:

- This symbol only works with point layers.
- *Outline* and *glowing* should not be used together; use one or the other.
- Not valid with ArcMap Server.

Notes:

None

Attribute descriptions for calloutMarkerSymbol:

Attribute	Usage
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
backColor	Background color using RGB values.
boundaryColor	Boundary color using RGB values.
font	Font name. The name is case sensitive. If font name uses "&", use "&" instead. For example, ESRI Transportation & Civic should be written as ESRI Transportation & Civic.
fontColor	Font color using RGB values.
fontSize	Font size.
fontStyle	Font style.
glowing	Glow color around text using RGB values.
interval	Distance between point and callout box; smaller number brings box closer to point.
outline	Outline color using RGB values.
shadow	Shadow color using RGB values.
transparency	Value to set percentage of transparency. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.

Examples for calloutMarkerSymbol:

Example 1:

```
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:layer layerId="5" visible="true">
    <aims:simpleLabelRenderer field="NAME">
      <aims:calloutMarkerSymbol font="Times New Roman" fontStyle="italic" fontSize="24"
fontColor="0,0,255" glowing="255,0,0" shadow="0,0,50" backColor="0,255,0" interval="10"
boundaryColor="255,255,0" transparency="0.8" antialiasing="false" />
    </aims:simpleLabelRenderer>
  </aims:layer>
</aims:map>
```

changeMetadataAccess

Used in: metadata
Parent elements: None

```
<aims:changeMetadataAccess
    Attributes that set values:
    private ="true | false"
    service ="string"

    Attributes that pass objects:
    connectionId ="object"
    docId ="string"
>

    No Child Elements
</aims:changeMetadataAccess >
```

Bold: Attribute or child element is required.

Description:

Changes the access of a metadata document.

Restrictions:

None

Notes:

None

Attribute descriptions for changeMetadataAccess:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
docId	String used to uniquely identify a document. The client used to publish the metadata is responsible for creating the document ID. This ID is automatically assigned when using ArcCatalog. If another client is used, aims:getUUID can be used to request a valid ID. The format for an ID is the following: {HHHHHHHH-HHHH-HHHH-HHHH-HHHHHHHHHHHH} where H is a hexadecimal digit (0–9,a–f,A–F). The ID is limited to 38 characters.
private	Determines whether document is viewable for all users. Use "false" if the document is viewable by all users. Use "true" when the document is viewable only by the document owner.
service	The name of the metadata service.

Examples for changeMetadataAccess:

Example 1:

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>changeMetadataAccess</title>
</head>
<body>
<aims:httpConnection id="myHttpConnection" url="http:\\localhost" username="publish"
password="publish" />

<aims:changeMetadataAccess connectionId="<%=(myHttpConnection)%>"
service="serviceMetadata" docId="{2D1DC1C2-14C1-4B39-81CF-47B7DABEA3B3}" private="false"
/>
</body>
</html>
```

changeOwner

Used in: metadata
Parent elements: None

```
<aims:changeOwner
    Attributes that set values:
    newOwner ="string"
    service ="string"

    Attributes that pass objects:
    connectionId ="object"
    docId ="string"
>

    No Child Elements
</aims:changeOwner >
```

Bold: Attribute or child element is required.

Description:

Allows a new owner to be assigned to a document. A document can only be edited by its owner.

Restrictions:

- In your access control list, you must have a role of "metadata_administrator", or the request will be denied.
- To add a new user (or owner), you must log into that Metadata Service as that user. To change an owner, it needs to be a value that you already used when logging in.

Notes:

None

Attribute descriptions for changeOwner:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
docId	String used to uniquely identify a document. The client used to publish the metadata is responsible for creating the document ID. This ID is automatically assigned when using ArcCatalog. If another client is used, aims:getUUID can be used to request a valid ID. The format for an ID is the following: {HHHHHHHH-HHHH-HHHH-HHHH-HHHHHHHHHHHH} where H is a hexadecimal digit (0–9,a–f,A–F). The ID is limited to 38 characters.
newOwner	New owner name. Must be a value that has already been used when logging in.
service	The name of the metadata service.

Examples for changeOwner:

Example 1:

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>changeOwner</title>
</head>
<body>
<aims:httpConnection id="myHttpConnection" url="http:\\localhost"
username="metadataadmin" password="metadataadmin" />

<aims:changeOwner connectionId="<%=(myHttpConnection)%>" service="serviceMetadata"
docId="{2D1DC1C2-14C1-4B39-81CF-47B7DABEA3B3}" newOwner="publish" />
</body>
</html>
```

chartSymbol

Used in: mapping
Parent elements: aims:exact aims:other aims:range aims:simpleLabelRenderer

```
<aims:chartSymbol
    Attributes that set values:
    antialiasing ="true | false"  [false]
    maxSize ="1 - NNN"
    maxValue ="1 - NNN"
    minSize ="1 - NNN"
    minValue ="1 - NNN"
    mode ="pie | bar"  [pie]
    outline ="0,0,0 - 255,255,255"  [none]
    shadow ="0,0,0 - 255,255,255"
    size ="1 - NNN"
    sizeField ="string"
    transparency ="0.0 - 1.0"  [1.0]
>
    (m) <chartValue... />
</aims:chartSymbol >
```

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Symbolizes features with bar or pie diagrams.

Although chart symbols draw on the map, no information about the chart symbols is displayed in the legend. ArcIMS 4 does not support displaying chart values in the legend.

Restrictions:

- The size of a chart can be determined in one of three ways:
    1. Using the *size* attribute.
    2. Using the *sizeField* attribute.
    3. Using the *minSize*, *maxSize*, *minValue*, and *maxValue* attributes.

  One of the three options must be used or no charts are drawn. *Size* ranks first in priority, and it takes precedence over the other attributes even if they are included. *SizeField* is second in priority, and it takes precedence over the other attributes if *size* is not present. *MinSize*, *maxSize*, *minValue*, and *maxValue* rank third in priority and are acknowledged only when *size* and *sizeField* are not present.
- Not valid with ArcMap Server.

Notes:

- A field listed in aims:chartSymbol *sizeField* must also be listed in aims:simpleLabelRenderer *field* or aims:valueMapLabelRenderer *labelField*.
- The attributes *minSize* and *maxSize* determine the minimum and maximum size of a chart, respectively. For each feature, the size of the chart is relative to other features based on the minimum and maximum values. These values are set using *minValue* and *maxValue*.

  If a feature has a value less than or equal to *minValue*, then the chart is drawn at *minSize*. If a feature has a value greater than or equal to *maxValue*, then the chart is drawn at *maxSize*. In the following example, the values are based on population. If a feature has a population less than or equal to 10,000, then the size of the chart is 10 pixels. If the feature has a population greater than or equal to 200,000, then the size of the chart is 50 pixels.

  <aims:chartSymbol minSize="10" minValue="10000" maxSize="50" maxValue="200000" >

  All other features are assigned a chart size between 10 and 50 pixels based on population. A feature with a population of 100,000 will have a chart approximately 30 pixels in size. A feature with a population of 1,000,000 is assigned a chart size of 50 pixels, since that is the maximum size allowed using *maxSize*.
- If the size assigned to a chart is large, the chart may not display. If you find that no charts are displaying, try a smaller chart size.

Attribute descriptions for chartSymbol:

Attribute	Usage
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
maxSize	Maximum size of chart if *size* or *sizeField* is not used.
maxValue	Maximum value that corresponds to the maximum chart size in *maxSize*.
minSize	Minimum size of chart if *size* or *sizeField* is not used.
minValue	Minimum value that corresponds to the minimum chart size in *minSize*.
mode	Type of chart: pie or bar.
outline	Outline color of charts using RGB values.
shadow	Shadow color using RGB values.
size	Size of charts. All charts are the same size.
sizeField	The field in the database containing the size of the chart. The field can be in the layer table or in a joined table. • For shapefiles with no joined tables, the field can be referenced using the short format. sizeField="AREA" • For shapefiles with joined tables, the name of the joined table must be included along with the field. sizeField="JOINEDTABLE.AREA" • For ArcSDE layers without joined tables, the field can be referenced using the short format. sizeField="AREA" The full long name can also be used. sizeField="ARCSDENAME.TABLE.AREA" • For ArcSDE layers with joined tables, joined fields must be referenced using the full long format. sizeField="ARCSDENAME.TABLE.AREA"
transparency	Value to set percentage of transparency. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.

Examples for chartSymbol:

Example 1:

```
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:layer layerId="5" visible="true">
    <aims:simpleLabelRenderer field="POP">
        <aims:chartSymbol size="15" mode="pie" >
          <aims:chartValue color="250,0,255" lookUpField="POP" />
        </aims:chartSymbol>
    </aims:simpleLabelRenderer>
  </aims:layer>
</aims:map>
```

chartValue

Used in: mapping
Parent elements: aims:chartSymbol

```
<aims:chartValue
    Attributes that set values:
    lookUpField ="string"
    color ="0,0,0 - 255,255,255"  [0,0,0]
    lower ="integer"
    upper ="integer"
    value ="integer"
>

    No Child Elements
</aims:chartValue >
```

Bold: Attribute or child element is required.

Description:

Sets which fields are used in pie and bar diagrams.

Although chart symbols draw on the map, no information about the chart symbols is displayed in the legend. ArcIMS 4 does not support displaying chart values in the legend.

Restrictions:

- The attribute *lookUpField* is not required when *value* is used. It is required in all other cases.
- Not valid with ArcMap Server.

Notes:

- All fields listed in aims:chartValue *lookUpField* must also be listed in aims:simpleLabelRenderer *field* or aims:valueMapLabelRenderer *labelField*.

- The attributes *lower* and *upper* are used together to change the color of a chart segment depending on the value for a feature. For example, a segment might be blue for population less than 1,000,000 and red for population greater than 1,000,000. For each range, a new aims:chartSymbol should be used for referencing the same *lookUpField* such as in the following example:

```
<aims:simpleLabelRenderer field="POP1999" >
  <aims:chartSymbol size="30" transparency="1.0" >
    <aims:chartValue lookUpField="POP1999" color="0,0,255" lower="0"
upper="1000000" />
    <aims:chartValue lookUpField="POP1999" color="255,0,0" lower="1000001"
upper="100000000" />
  </aims:chartSymbol>
</aims:simpleLabelRenderer>
```

- The attribute *value* is used to change the color of a chart segment depending on user-assigned integer values. All values within the aims:chartSymbol group are a ratio of the sum of the values. For example, assume two aims:chartValue tags are used to build a chart, and the values are 14018000 and 15743000. The sum of these values is 29761000. Therefore, the size of the first chart segment is 14018000 / 29761000, or 47 percent of the chart. The second segment is 53 percent.

  In the following example, values were calculated ahead of time for populations under 30 and 30 and over for California. These two values were then assigned to two aims:chartValue tags. The aims:valueMapLabelRenderer *labelField* must be present even though it is not used. For the string to parse properly, a valid field name in *labelField* must be included. If aims:simpleLabelRenderer were used, *field* would be used instead. Again, a valid field name must be included. In aims:chartValue, no *lookUpField* is needed.

```
<aims:valueMapLabelRenderer lookUpField="STATE_NAME" labelField="POP1999" >
  <aims:exact value="California">
    <aims:chartSymbol size="50" >
      <aims:chartValue color="255,0,0" value="14018000" />
      <aims:chartValue color="0,0,255" value="15743000" />
    </aims:chartSymbol>
  </aims:exact>
</aims:valueMapLabelRenderer>
```

- The *lookUpField* order must be the same order as fields listed in aims:simpleLabelRenderer *field* or aims:valueMapLabelRenderer *labelField*.

Attribute descriptions for chartValue:

Attribute	Usage
color	Color used for segment of pie or bar chart using RGB values.
lookUpField	Lookup field for chart segment. The field can be in the layer table or a joined table. • For shapefiles with no joined tables, the field can be referenced using the short format. lookUpField="AREA" • For shapefiles with joined tables, the name of the joined table must be included along with the field. lookUpField="JOINEDTABLE.AREA" • For ArcSDE layers without joined tables, the field can be referenced using the short format. lookUpField="AREA" The full long name can also be used. lookUpField="ARCSDENAME.TABLE.AREA" • For ArcSDE layers with joined tables, joined fields must be referenced using the full long format. lookUpField="ARCSDENAME.TABLE.AREA"
lower	Used together with *upper* to determine the minimum value for drawing a chart segment.
upper	Used together with *lower* to determine the maximum value for drawing a chart segment.
value	Used to change the color of a chart segment depending on user-assigned integer values. All values within the aims:chartSymbol group are a ratio of the sum of the values.

Examples for chartValue:

Example 1:

```
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:layer layerId="5" visible="true">
    <aims:simpleLabelRenderer field="POP">
        <aims:chartSymbol size="15" mode="pie" >
          <aims:chartValue color="250,0,255" lookUpField="POP" />
        </aims:chartSymbol>
    </aims:simpleLabelRenderer>
  </aims:layer>
</aims:map>
```

coordSys

Used in: mapping
Parent elements: aims:acetateObject aims:createLayer aims:getRasterInfo

```
<aims:coordSys
    Attributes that set values:

    When using ArcMap Server:
    id ="integer"
    string ="string"

    When using Image Server:
    datumTransformId ="integer"
    datumTransformString ="string"
    id ="integer"
    string ="string"
>

    No Child Elements
</aims:coordSys >
```

Description:

Defines the projection coordinate system of a layer. Aims:coordSys cannot be used to project a layer; its purpose is to provide the metadata for the layer. aims:featureCoordSys and aims:filterCoordSys are used to project the layers to a specified projection.

Restrictions:

- Must use either *id* or *string*, but not both.
- For datum transformations either *datumTransformId* or *datumTransformString* is used, but not both.
- In ArcMap Image Services, *datumtransformid* or *datumtransformstring* are not valid.

Notes:

- For a complete list of supported IDs and definition strings, see:
    - Projected Coordinate Systems Listing [Sorted by projection ID] [Sorted by name]
    - Geographic Coordinate Systems Listing [Sorted by projection ID] [Sorted by name]
    - Datum Transformation Listing [Sorted by projection ID] [Sorted by name]
- When using definition strings, the quotes in the string must be preceded by a backslash (\) so both the Java Connector and the ArcIMS Spatial Server can interpret the string correctly. For example, the definition string for World Mollweide is:

    PROJCS["World_Mollweide",GEOGCS["GCS_WGS_1984",DATUM["D_WGS_1984",SPHEROID["WGS_1984",6378137,298.257223563]],PRIMEM["Greenwich",0],UNIT["Degree",0.0174532925199433295]],PROJECTION["Mollweide"],PARAMETER["False_Easting",0],PARAMETER["False_Northing",0],PARAMETER["Central_Meridian",0],UNIT["Meter",1]]

    Once the string is modified, it should look like this:

    PROJCS[\"World_Mollweide\",GEOGCS[\"GCS_WGS_1984\",DATUM[\"D_WGS_1984\",SPHEROID[\"WGS_1984\",6378137,298.257223563]],PRIMEM[\"Greenwich\",0],UNIT[\"Degree\",0.0174532925199433295]],PROJECTION[\"Mollweide\"],PARAMETER[\"False_Easting\",0],PARAMETER[\"False_Northing\",0],PARAMETER[\"Central_Meridian\",0],UNIT[\"Meter\",1]]

    Note that the syntax for escaping quotes with the Java Connector is different for the Servlet Connector.
- The attributes *datumTransformId* and *datumTransformString* are used when datum transformation information needs to be included. Only datum transformations to and from WGS 1984 are supported.
    - When these attributes are used with aims:coordSys and aims:filterCoordSys, the datum transformation is from a non-WGS 1984 datum to WGS 1984. For example, Pulkovo_1942_To_WGS_1984 (datumtransformid="8157") transforms data from Pulkovo 1942 to WGS 1984.
    - When these attributes are used with aims:featureCoordSys, the datum transformation is from WGS 1984 to a non-WGS 1984 datum. In the above example, the datum transformation is from WGS 1984 to Pulkovo 1942.
- When using aims:coordSys in an acetate layer, it should be placed inside acetateObject rather than aims:createLayer.
- aims:coordSys can be used to designate the current projection of an image. Image layers can be projected along with feature and acetate layers.
- aims:coordSys is not needed if no datum transformation information is needed for the layer and:
    - A *.prj file is available for a shapefile.

- A *.prj file is available for a coverage in ArcSDE for Coverages.
- A spatial references table is used in ArcSDE.

Datum transformation information is not included in *.prj files and spatial reference tables. Therefore, aims:coordSys must be included if a datum transformation is needed.

- If a layer does not project, double-check that a *.prj file or spatial reference table exists for the layer. If not, aims:coordSys must be included.

Attribute descriptions for coordSys:

Attribute	Usage
datumTransformId	Datum transformation ID.
datumTransformString	Datum transformation definition string.
id	Projected or geographic coordinate system ID.
string	Projected or geographic coordinate system definition string.

Examples for coordSys:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myConnection%>"
name="imageServiceName"/>

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">

  <aims:getRasterInfo id="myRaster" layerId="0" x="1099.067" y="1349.42">
    <aims:coordSys id="4326" />
  </aims:getRasterInfo>

  <aims:iterateRasterInfo rasterInfo="<%=(myRaster)%>" rasterId="id" number="no"
value="val">
    <b>Raster Id = </b> <%=(id)%>
    <b>Number    = </b> <%=(no)%>
    <b>Value     = </b> <%=(val)%><br>
  </aims:iterateRasterInfo>
</aims:map>
```

createDocument

Used in: metadata
Parent elements: None

<aims:createDocument

Attributes that set values:
name ="*string*"
content ="liveData | downloadableData | offlineData | staticMapImage | document | application | geographicService | clearinghouse | mapFiles | geographicActivities | unknown"
documentContent ="*string*"
folder ="true | false" **[false]**
maxx ="*double*"
maxy ="*double*"
minx ="*double*"
miny ="*double*"
onlink ="*string*"
parentDocId ="*string*"
private ="true | false" **[false]**
server ="*string*"
service ="*string*"
serviceType ="metadata | image | feature | wms"
thumbnail ="*string*"

Attributes that pass objects:
docId ="*string*"

Attributes that return values:
id ="*object*"
>

No Child Elements
</aims:createDocument >

Bold: Attribute or child element is required.

Description:

Publishes a metadata document from a client to the server.

Restrictions:

None

Notes:

The Document object returned by aims:createDocument can be used in aims:publishDocument and aims:deleteDocument.

Attribute descriptions for createDocument:

Attribute	Usage
content	The values for this attribute correspond to the content t in the dropdown comb box of the Metadata Explorer.
docId	String used to uniquely identify a document. The client used to publish the metadata is responsible for creating the document ID. This ID is automatically assigned when using ArcCatalog. If another client is used, aims:getUUID can be used to request a valid ID. The format for an ID is the following: {HHHHHHHH-HHHH-HHHH-HHHH-HHHHHHHHHHHH} where H is a hexadecimal digit (0–9,a–f,A–F). The ID is limited to 38 characters.
documentContent	The content of the XML document associated with this dataset as a string.
folder	Determines whether document represents a folder. Use "false" if the document does not contain other documents. Use "true" if the document represents a folder.

Tag Attribute Value	Comb	ntent
liveData	Live D	laps
downloadableData	Down	)ata
offlineData	Offline	
staticMapImage	Static	es
documents	Other	ts
application	Applic	
geographicService	Geographic Services	
clearinghouse	Clearinghouses	

id	Returns the Document object.
maxx	Maximum x-coordinate in map units.
maxy	Maximum y-coordinate in map units.
minx	Minimum x-coordinate in map units.
miny	Minimum y-coordinate in map units.
name	Name that identifies the dataset corresponding to the document.
onlink	A string identifying the location of the dataset corresponding to the document.
parentDocId	Unique string for identifying a document belonging to the parent name and parent owner.
private	Determines whether document is viewable for all users. Use "false" if the document is viewable by all users. Use "true" when the document is viewable only by the document owner.
server	The URL of the Web server containing data or metadata associated with the document, for example, http://mymachine.domain.com.
service	The name of the ArcIMS service containing data or metadata associated with the document.
serviceType	Type of ArcIMS service.
thumbnail	URL of the thumbnail image.

Examples for createDocument:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true"
username="publish" password="publish" />

<aims:getUUID id="myUUID" connectionId="<%=(myConnection)%>" service="serviceName" />
<%
String docId = myUUID.elementAt(0).toString();
%>

<aims:createDocument id="myDocument" docId="<%=(docId)%>" name="Document" folder="true"
/>

<aims:publishDocument connectionId="<%=(myConnection)%>" document="<%=(myDocument)%>"
service="serviceName" />
```

createLayer

Used in: mapping
Parent elements: aims:map

<aims:createLayer

Attributes that set values:
layerId ="*string*"
type ="shape | sde | acetate | image"
name ="*string*"
visible ="true | false" **[true]**
>

Nested tag for defining an acetate object:
<acetateObject... />

Nested tag for referencing a workspace:
<dataset... />

Nested tags for defining a renderer:
<groupRenderer... /> *[Or]*
<scaleDependentRenderer... /> *[Or]*
<simpleLabelRenderer... /> *[Or]*
<simpleRenderer... /> *[Or]*
<valueMapLabelRenderer... /> *[Or]*
<valueMapRenderer... /> *[Or]*

Nested tags for defining a workspace:
<imageWorkspace... /> *[Or]*
<sdeWorkspace... /> *[Or]*
<shapeWorkspace... /> *[Or]*

Nested tags for identifying the coordinate system of the layer:
<coordSys... />
<densify... />

Nested tags for showing a subset of data:
<displayFeatures... />
<filter... />

</aims:createLayer >
Bold: Attribute or child element is required.

Description:

Adds a dynamic layer to a map.

Restrictions:

Only one renderer and/or one workspace as a subtag of aims:createLayer may be used.

Notes:

To add a shapefile, image, or ArcSDE layer, <MAP dynamic="true" > must be set in the map configuration file.

Attribute descriptions for createLayer:

Attribute	Usage
layerId	Unique ID for a layer. The ID can be any combination of alpha and numeric characters.
name	Layer name. Can be an alias.
type	Specifies layer type. Use "featureclass" for shapefiles and ArcSDE vector layers. Use "image" for raster image files, GRIDs, and ArcSDE raster layers. Use "acetate" for adding graphics on top of the map.
visible	Specifies layer visibility.

Examples for createLayer:

Example 1:

```
<aims:tcpConnection id="mytcpConnection" port="5300" host="myMachine" />

<aims:mapService id="myService" connectionId="<%=mytcpConnection%>" name="mexico" />

<aims:map id="mapImage" serviceId="<%=myService%>" width="200" height="200" >

<aims:createLayer layerId="111" type="acetate" visible="true">
  <aims:acetateObject units="pixel">
    <aims:scalebar x="10" y="10" fontSize="5" barWidth="100" mapUnits="decimal_degrees"
scaleUnits="miles" distance="10.0" barTransparency="1" />
  </aims:acetateObject>
</aims:createLayer>

</aims:map>
```

createService

Used in: administration
Parent elements: None

```
<aims:createService

    Attributes that set values:
    configContents ="string | InputStream | ByteArray"
    configFile ="string"
    imageType ="jpg | gif | png | png8 | bmp"  [jpg]
    name ="string"
    outputDir ="string"
    outputURL ="string"
    pixelCount ="integer"  [480000]
    virtualServer ="string"
    virtualServerType ="FeatureServer | ImageServer | MetadataServer"
    encode ="true | false"  [false]
    outputCleanup ="integer"  [10]
    start ="true | false"  [false]
    virtualServerVersion ="ArcMap"  [""]

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    id ="object"
    error ="string"
>

    No Child Elements
</aims:createService >
```

Bold: Attribute or child element is required.

Description:

Creates an ArcIMS service.

Restrictions:

GIF format is not supported as an output format with the ArcMap Server. GIF format can be used with the Image Server.

Notes:

None

Attribute descriptions for createService:

Attribute	Usage
configContents	Sets the contents of the configuration file.
configFile	The location of the configuration file for this service.
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
encode	If "true", sets the encoding of the configuration contents to base64.
error	Returns an error string if any occurred. Returns "Success" if no error.
id	Returns the Service object created.
imageType	Defines the type of image output for a given service. GIF format is not valid with ArcMap Image Services.
name	The name of the service to be created.
ouputDir	Sets the output directory for image, extract, and other services that output files.
outputCleanup	The cleanup interval for files generated by service.
outputURL	Sets the output URL for image, extract, and other services that output files.
pixelCount	Represents the maximum number of pixels allowed in a map image. The calculation is made by multiplying the width times the height. For example, an image 400 x 600 in size contains 240,000 pixels.

start	If "true", starts the service after it is created.
virtualServer	The name of the Virtual Server associated with this service.
virtualServerType	The type of virtual server.
virtualServerVersion	Sets the version string for the Virtual Server.

Examples for createService:

Example 1: Using fileUpload bean (as implemented in Web Administrator).

```
<jsp:useBean id="TheBean" scope="page" class="com.esri.aims.mtier.model.util.FileUpload"
/>

<%
TheBean.doUpload(request);
String name = TheBean.getFieldValue("name");
String vserver = TheBean.getFieldValue("vserver");
if (description==null) description = "";
String outfile = TheBean.getFieldValue("outfile");
if (outfile==null) outfile = "";
String outurl = TheBean.getFieldValue("outurl");
if (outurl==null) outurl = "";
String imagetype = TheBean.getFieldValue("imagetype");
if (imagetype==null) imagetype = "jpeg";
String pixelcount = TheBean.getFieldValue("pixelcount");
if (pixelcount==null) pixelcount = "4";
int pixel = (Integer.parseInt(pixelcount) * 262144);
String pixelStr = Integer.toString(pixel);
String cleanupStr = TheBean.getFieldValue("cleanup");
if (cleanupStr==null) cleanupStr = "0";

String configFile = TheBean.getFilepath() + TheBean.getFilename();
InputStream configStream = TheBean.getUploadedFileAsStream();
%>

<html>
```

```
<head>
      <title>Adding Service - Administrator</title>
</head>

<body bgcolor="#336699" text="White" link="#000099" vlink="#000099" alink="#999999" >
<div align="center"><font face="Arial,Helvetica,sans-serif"><b>

<aims:httpConnection
      id="myhttpConnection"
      url="http://mymachine"
      debug="true"
        role="admin"
      username="myUsername"
      password="myPassword"
/>

<aims:createService id="myService" connectionId="<%=myhttpConnection%>"
name="<%=name%>"configFile="<%=configFile%>" configContents="<%=configStream%>"
virtualServer="<%=vserver%>" virtualServerType="<%=vsType%>"
virtualServerVersion="<%=vsVersion%>" imageType="<%=imagetype%>"
outputCleanup="<%=cleanupStr%>" outputDir="<%=outfile%>" outputURL="<%=outurl%>"
pixelCount="<%=pixelStr%>" encode="<%=doEncode%>" error="myError" start="true" />
<%

//
if (TheBean.getUploadedFile().length()>0) {
} else { %>
            Unable to upload file.

<% } %>
```

createVirtualServer

Used in: **administration**
Parent elements: None

```
<aims:createVirtualServer

    Attributes that set values:
    containers ="string, string, ..."
    name ="string"
    type ="FeatureServer | ImageServer | MetadataServer"
    access ="private | public"  [public]
    description ="string"  [""]
    recyclingFrequency ="1 | 2 | 3 | 4 | 6 | 12 | 24"  [2]
    recyclingHour ="0 - 23"  [0]
    recyclingMinute ="0 | 15 | 30 | 45"  [0]
    threadsForContainers ="integer, integer, .."  [2,2, ..]
    version ="ArcMap | """  [""]

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    id ="string"
    error ="string"
>

    No Child Elements
</aims:createVirtualServer >
```

Bold: Attribute or child element is required.

Description:

Creates a virtual server.

Restrictions:

When *version="ArcMap"*, the values for *threadForContainers* must be "1" for each container. If you use any value other than "1", an error message will be returned stating "Attempting to start multiple threads for a single-threaded server group".

Notes:

The attributes *containers* and *threadForContainers* have a one-to-one correlation within the comma-delimited lists. The order of items should be the same in both lists.

Attribute descriptions for createVirtualServer:

Attribute	Usage
access	Determines whether public or private access is allowed to this Virtual Server.
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
containers	A comma-delimited list of ArcIMS Spatial Servers to add to this Virtual Server.
description	The Virtual Server description.
error	Returns an error string if any occurred. Returns "Success" if no error.
id	Returns the VirtualServer object created.
name	The name of the new Virtual Server.
recyclingFrequency	Number of times per each 24-hour period to recycle all Spatial Servers associated with this Virtual Server. If this value is 0, the Virtual Server does not implement recycling. • 1—Once a day • 2—Twice a day • 3—Three times a day • 4—Four times a day • 6—Every 4 hours • 12—Every 2 hours • 24—Every hour

recyclingHour	Starting hour to begin the first recycling period each day using a 24-hour clock. "0" is midnight, and "23" is 11:00 PM.
recyclingMinute	Starting minute to begin the first recycling period each day in 15-minute increments.
threadsForContainers	The comma-delimited list of number of instances (threads) to associate with each container listed in the *container* attribute. If the number of threads is not specified, the default is 2. When *version="ArcMap"*, the values for *threadForContainers* must be "1" for each container. If you use any value other than "1", an error message will be returned stating "Attempting to start multiple threads for a single-threaded server group".
type	The type of Virtual Server to be created.
version	The Virtual Server version.

Examples for createVirtualServer:

Example 1:

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://mymachine"
      role="admin"
      debug="true"
      username="admin"
      password="admin"
/>

<aims:createVirtualServer id="myNewServer" connectionId="<%=myhttpConnection%>"
name="ImageServerArcMap2" type="ImageServer" version="ArcMap"
containers="mymachine_1,mymachine_2,mymachine_3" threadsForContainers="1,1,1"
error="myError" />

<aims:iterateVirtualServers virtualServers="<%=myNewServer%>" count="serverCount" >
      <aims:getVirtualServerAttribute id="name" attribute="NAME" />
      <aims:getVirtualServerAttribute id="type" attribute="TYPE" />
      <aims:getVirtualServerAttribute id="description" attribute="DESCRIPTION" />
      <aims:getVirtualServerAttribute id="version" attribute="VERSION" />
      <aims:getVirtualServerAttribute id="access" attribute="ACCESS" />
```

```
      <aims:getVirtualServerAttribute id="containerset" attribute="CONTAINERSET" />
      <td>
        <br><b>name is: <%=name%></b>
        type is: <%=type%>
        description is: <%=description%>
        version is: <%=version%>
        access is: <%=access%>
        containerset is: <%=containerset%>
      </td>
</aims:iterateVirtualServers>
```

dataframe

Used in: mapping
Parent elements: aims:getArcMapLayout

```
<aims:dataframe
    Attributes that set values:
    dataframeName ="string"
>

    <featureCoordSys... />
    <filterCoordSys... />
    (m) <layer... />
    <layerList... />
    <envelope... />  [Or]
    <scale... />  [Or]

</aims:dataframe >
```

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Provides framework for listing properties of a specified data frame in an ArcMap layout.

Restrictions:

- Either aims:envelope or aims:scale may be used, but not both. If both are present, aims:scale takes precedence.
- If multiple data frames in the ArcMap document have the same name, only the first data frame with the name is drawn.
- Valid with ArcMap Server only.

Notes:

None

Attribute descriptions for dataframe:

Attribute	Usage
dataframeName	Name of data frame as defined in ArcMap document.

Examples for dataframe:

Example 1:

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>Layer nested inside Dataframe tag example</title>
<LINK STYLE="text/css" REL="stylesheet" HREF="font.css">
</head>
<body>
<aims:tcpConnection id="myConnection" host="localhost" port="5300" debug="true" />

<aims:arcMapService id="arcmyDataframeCollection" connectionId="<%=(myConnection)%>"
name="multidf" error="arcmyError" />

<aims:getArcMapLayout dataframeCollection="<%=(arcmyDataframeCollection)%>" width="500"
height="700">
  <aims:envelope minx="0" miny="0" maxx="50" maxy="40" />
  <aims:layoutOutput type="jpg" />
  <aims:dataframe dataframeName="South East Asia">
    <aims:layer layerId="1" visible="false" />
    <aims:layer layerId="2" visible="false" />
    <aims:layer layerId="3" visible="false" />
    <aims:featureCoordSys id="102030" />
    <aims:filterCoordSys id="102030" />
```

```
    <aims:scale rf="3251279" x="-5079159.83" y="2146309.62"  />
  </aims:dataframe>
  <aims:dataframe dataframeName="Europe">
    <aims:layer layerId="2" visible="true" />
    <aims:layer layerId="0" visible="false" />
    <aims:envelope minx="19.8889685482977" miny="52.7973466116867"
maxx="35.9893558857411" maxy="63.792074206216" />
  </aims:dataframe>
</aims:getArcMapLayout>

<aims:getArcMapLayoutAttribute id="arclayoutURL" value="LAYOUTURL"
dataframeCollection="<%=(arcmyDataframeCollection)%>" />
<img src="<%=(arclayoutURL)%>" /><br>

</body>
</html>
```

dataset

Used in: mapping
Parent elements: aims:createLayer

```
<aims:dataset
    Attributes that set values:
    name ="string"
    type ="point | line | polygon | image"
    workspace ="string"
>

    No Child Elements
</aims:dataset >
```

Bold: Attribute or child element is required.

Description:

Defines the dataset used in the layer.

Restrictions:

None

Notes:

There are five different ways to access images and GRIDs as detailed in the table below.

Image Access Method	Workspace Location	DATASET Layer Name
Specify by name.	Use aims:imageWorkspace; attribute *directory* points to location of specified image.	Name of image including its extension.
Use a ll images in a directory. Images in the same directory automatically tile if they use the same coordinate projection and are drawn when they are within the extent requested.	ims:imageWorkspace; attribute *ory* points to location of the group of images.	*ImageDirectory for the name: ne="*ImageDirectory"
n ArcView image catalog.	ims:imageWorkspace; attribute *directory* points to location of catalog, not images.	Na ne of the image catalog DBF file. For instance, if catalog is named imagecat.dbf, use name="imagecat.dbf".
GRID.	Use a ims:imageWorkspace. A GRID has tv o directories: one for the GRID data and one for the INFO files. Both these directories should be grouped toget her under a parent directory. The *irectory* attribute points to the parer t directory above the GRID and INFO directories. A *.clr file can be inclu ed to color the GRID. This file shoul d have the same name as the GRID and be included in the parent directory above the INFO and GRID directories.	ne of directory that contains ID data; for a GRID named zard, use name="blizzard".
n image in ArcSDE.	ims:sdeWorkspace.	Na ne is the full ArcSDE name inc uding the field name where the ima ge resides. In this example, "REDLANDS" is the field where the image resides: RASTER.IMAGES.REDLANDS.

Attribute descriptions for dataset:

Attribute	Usage
name	For shapefiles, use the name of the data file without an extension, such as STATES. For ArcSDE, use the full name of the layer such as DATA.STATES. For images, refer to the table in the Notes section.
type	Source layer feature type.
workspace	References the workspace name where the data resides.

Examples for dataset:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />
<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName"/>

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
<aims:createLayer layerId="topoq" type="shape" name="topoquad" visible="true" >
  <aims:simpleRenderer >
    <aims:simpleLineSymbol color="0,255,0" width="1" />
  </aims:simpleRenderer>
  <aims:shapeWorkspace name="workspace1"
directory="C:\\ArcIMS\\AXL\\TutorialData\\SantaClara" featureClass="line" />
  <aims:dataset name="sc_topoq24" type="polygon" workspace="workspace1" />
</aims:createLayer>
</aims:map>

<img src="<%=(myMap)%>" />
```

datasetProperty

Used in: metadata
Parent elements: aims:iterateMetadata

```
<aims:datasetProperty
    Attributes that set values:
    value ="THUMBNAIL | NAME | GND | URL | ONLINK | OWNER | SERVER | SERVICE | SERVICETYPE |
TIMESTAMP | DOCID | ENVELOPE | MINX | MINY | MAXX | MAXY | PARENT | FOLDER | CONTENT | INDEXSTATUS
| CHILDREN | PRIVATE | REFCOUNT | UPDATED"
    dataset ="string"

    Attributes that return values:
    id ="string"
>

    No Child Elements
</aims:datasetProperty >
```

Bold: Attribute or child element is required.

Description:

Used to get individual items from the results of searching or browsing the metadata repository.

Restrictions:

The attribute *dataset* is not required when nested inside aims:iterateMetadata.

Notes:

Column names added through RESPONSE_COLUMN in the map configuration file become "custom attributes" in aims:datasetProperty. For example, if you include RESPONSE_COLUMN *columnname="service_running"*, then in addition to the existing attributes in aims:datasetProperty, a new value named SERVICE_RUNNING can be specified in the aims:datasetProperty *value* attribute.

<aims:datasetProperty id="service_running" value="SERVICE_RUNNING" />

Attribute descriptions for datasetProperty:

Attribute	Usage
dataset	The metadata document to retrieve the information from.
id	Returns the value of the property specified in the *value* attribute.
value	The name of the document property to retrieve.

Examples for datasetProperty:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300"/>

<aims:search  id="searchResults" connectionId="<%=myConnection%>" service="serviceName"
sort="name" operator="and">
  <aims:documentInfo name="Parent" />
</aims:search>

<h1>Search Result Size : <%=(searchResults.size())%></h1>

<aims:iterateMetadata iterate="<%=searchResults%>" size="mySize" >
  <aims:datasetProperty id="myDocId" value="DOCID" />
  Document Id : <%=(myDocId)%><br>
  <aims:datasetProperty id="myFolder" value="FOLDER" />
  Folder      : <%=(myFolder)%><br>
  <aims:datasetProperty id="myUpdated" value="UPDATED" />
  Updated     : <%=(myUpdated)%><br>
</aims:iterateMetadata>
```

deleteDocument

Used in: metadata
Parent elements: None

```
<aims:deleteDocument
    Attributes that set values:
    docId ="string"
    service ="string"

    Attributes that pass objects:
    connectionId ="object"
>

    No Child Elements
</aims:deleteDocument >
```

Bold: Attribute or child element is required.

Description:

Deletes a metadata document from the metadata repository.

Restrictions:

This element can be used only by the owner established in aims:createDocument. Any other user will get an error message.

Notes:

None

Attribute descriptions for deleteDocument:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
docId	String used to uniquely identify a document. The client used to publish the metadata is responsible for creating the document ID. This ID is automatically assigned when using ArcCatalog. If another client is used, aims:getUUID can be used to request a valid ID. The format for an ID is the following: {HHHHHHHH-HHHH-HHHH-HHHH-HHHHHHHHHHHH} where H is a hexadecimal digit (0–9,a–f,A–F). The ID is limited to 38 characters.
service	The name of the metadata service containing the document to delete.

Examples for deleteDocument:

Example 1: Deletes a metadata document from the metadata repository.

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true"
username="publish" password="publish" />

<aims:deleteDocument connectionId="<%=(myConnection)%>" docId="{d78914f0-1a61-11d6-bd72-
f713332c6e33}" service="serviceName" />
```

deleteMetadataRelationship

Used in: metadata
Parent elements: None

```
<aims:deleteMetadataRelationship

    Attributes that set values:
    service ="string"

    Attributes that pass objects:
    connectionId ="object"
    docId ="string"
>

    (m) <metadataChild... />  [And/Or]
    (m) <metadataSibling... />  [And/Or]

</aims:deleteMetadataRelationship >
```

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Deletes relationships beween a source metadata document and one or more child or sibling metadata documents.

Restrictions:

None

Notes:

A child document is a subdocument to the current metadata document. A sibling document is a related document. When selected as a sibling, the document is listed under the parent document under the Related Documents link.

Attribute descriptions for deleteMetadataRelationship:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
docId	String used to uniquely identify a document. The client used to publish the metadata is responsible for creating the document ID. This ID is automatically assigned when using ArcCatalog. If another client is used, aims:getUUID can be used to request a valid ID. The format for an ID is the following: {HHHHHHHH-HHHH-HHHH-HHHH-HHHHHHHHHHHH} where H is a hexadecimal digit (0–9,a–f,A–F). The ID is limited to 38 characters.
service	The name of the metadata service.

Examples for deleteMetadataRelationship:

Example 1:

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>deleteMetadataRelationship</title>
</head>
<body>
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" username="publish"
password="publish" />

<aims:deleteMetadataRelationship connectionId="<%=(myConnection)%>" service="serviceName"
docId="{E6F7632B-04CE-4B50-A995-887DD144F4DC}">

  <aims:metadataChild docId="{9F582D73-7001-4A4B-A806-0FB8392E1F8F}" />
  <aims:metadataSibling docId="{C7F0DEBF-BCF6-48E7-847C-72C31A1DBCBD}" />

</aims:deleteMetadataRelationship>
</body>
</html>
```

densify

Used in: mapping
Parent elements: aims:createLayer

```
<aims:densify
    Attributes that set values:
    tolerance ="double"
>

    No Child Elements
</aims:densify >
```

Bold: Attribute or child element is required.

Description:

The process of data densification adds virtual points to a layer before the layer is projected.

Restrictions:

None

Notes:

- Using aims:densify in a request can slow down the ArcIMS Spatial Server noticeably. Use only when features are missing from one or more layers in a requested image, usually around the periphery. It is better to start with a large number for *tolerance*. A guideline is to start with a value about 20 to 30 percent of the distance between the minimum and maximum x-coordinates of the current map extent.
- The units for *tolerance* are the same as the units for the layer. For example, if the layer is in feet, the tolerance distance is in feet.
- Needed only if a layer is going to be projected. If the layer is in the same projection as the service, aims:densify does not need to be used.

Attribute descriptions for densify:

Attribute	Usage
tolerance	Defines distance (tolerance) between points and is applied on geometry before projecting takes place.

Examples for densify:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName"/>

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">

<aims:createLayer layerId="topoq" type="shape" name="topoquad" visible="true" >
  <aims:coordSys id="26744" />
  <aims:densify tolerance="100" />

  <aims:shapeWorkspace name="workspace1"
directory="C:\\ArcIMS\\AXL\\TutorialData\\SantaClara" featureClass="line" />
  <aims:dataset name="sc_topoq24" type="polygon" workspace="workspace1" />
  <aims:simpleRenderer >
    <aims:simpleLineSymbol color="255,0,250" width="12" />
  </aims:simpleRenderer>
</aims:createLayer>

</aims:map>
<img src="<%=(myMap)%>" />
```

displayFeatures

Used in: mapping
Parent elements: aims:createLayer aims:layer

```
<aims:displayFeatures

    Attributes that set values:
    expression ="string"
    relation ="area_intersection | envelope_intersection"  [area_intersection]
    whereExpression ="string"
    zoomToFeatures ="true | false"  [false]
>


    One spatial object tag can be selected:
    <envelope... />  [Or]
    <line... />  [Or]
    <point... />  [Or]
    <polygon... />  [Or]

    One symbol tag can be selected:
    <gradientFillSymbol... />  [Or]
    <hashLineSymbol... />  [Or]
    <rasterFillSymbol... />  [Or]
    <rasterMarkerSymbol... />  [Or]
    <simpleLineSymbol... />  [Or]
    <simpleMarkerSymbol... />  [Or]
    <simplePolygonSymbol... />  [Or]
    <truetypeMarkerSymbol... />  [Or]

</aims:displayFeatures >
```

Description:

Selects and zooms to a specified feature or group of features on the map.

Restrictions:

Only one symbol and one spatial object tag can be used at a time.

Notes:

None

Attribute descriptions for displayFeatures:

Attribute	Usage
expression	Expression to query for the desired features. This attribute has been deprecated in ArcIMS 4.0.1. Use *whereExpression* instead.
relation	Describes spatial relation. See Notes section under aims:filter for more details.
whereExpression	Expression to query for the desired features.
zoomToFeatures	Specifies whether to zoom to the selected features.

Examples for displayFeatures:

Example 1: When using the whereExpression attribute.

```
<%@page contentType="text/html"%>
<%@taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>displayFeatures with whereExpression</title>
</head>
<body>
<aims:tcpConnection id="myTCPConnection"  host="localhost" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="SantaClara"
/>

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
```

```
<aims:layer layerId="5" visible="true">
  <aims:displayFeatures whereExpression="NAME='San Jose'" zoomToFeatures="false">
    <aims:simpleMarkerSymbol width="25" type="star" color="0,0,255" outline="0,0,0"
shadow="0,0,0" />
  </aims:displayFeatures>
</aims:layer>

</aims:map>
<img src="<%=(myMap)%>" />
</body>
</html>
```

Example 2: When using a spatial object.

```
<%@page contentType="text/html"%>
<%@taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>displayFeatures with Spatial Object</title>
</head>
<body>
<aims:tcpConnection id="myTCPConnection"  host="localhost" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="SantaClara"
/>

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">

<aims:layer layerId="3" visible="true">
  <aims:displayFeatures zoomToFeatures="false" relation="envelope_intersection">
    <aims:simpleLineSymbol transparency="1.0" type="dash" width="5" capType="round"
joinType="round" color="0,0,255"  />
    <aims:envelope minx="-121.98" miny="37.398" maxx="-121.823" maxy="37.265" />
  </aims:displayFeatures>
</aims:layer>
```

```
</aims:map>
<img src="<%=(myMap)%>" />
</body>
</html>
```

Example 3: When using whereExpressions and a spatial object.

```
<%@page contentType="text/html"%>
<%@taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>displayFeatures with Spatial Object and expression</title>
</head>
<body>
<aims:tcpConnection id="myTCPConnection"  host="localhost" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="SantaClara"
/>

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">

<aims:layer layerId="5" visible="true">
  <aims:displayFeatures whereExpression="POP >= 93613" zoomToFeatures="false"
relation="envelope_intersection">
    <aims:simpleMarkerSymbol width="25" type="star" color="0,0,255" outline="0,0,0"
shadow="0,0,0" />
    <aims:envelope minx="-121.98" miny="37.398" maxx="-121.823" maxy="37.265" />
  </aims:displayFeatures>
</aims:layer>

</aims:map>
<img src="<%=(myMap)%>" />
</body>
</html>
```

documentInfo

Used in: metadata
Parent elements: aims:nestedSearch aims:search

```
<aims:documentInfo
    Attributes that set values:
    name ="string"
    owner ="string"
    content ="liveData | downloadableData | offlineData | staticMapImage | document | application | geographicService |
clearinghouse | mapFiles | geographicActivities"
>

    No Child Elements
</aims:documentInfo >
```

Description:

Specifies the name, owner, and content type of a document. With this information, documents can be searched by a specific name, content type, or a certain user.

Restrictions:

None

Notes:

None

Attribute descriptions for documentInfo:

Attribute	Usage
content	Document content type.
name	Name that identifies the dataset corresponding to the document.
owner	Name identifying the owner of the metadata document.

Examples for documentInfo:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true"
username="publish" password="publish" />

<aims:search  id="searchResults" connectionId="<%=myConnection%>" service="serviceName"
sort="name" operator="and">

  <aims:documentInfo name="Parent" />

</aims:search>

<h1>Search Result Size : <%=(searchResults.size())%></h1>
```

envelope

Used in: mapping
Parent elements: aims:arcMapService aims:dataframe aims:displayFeatures aims:filter aims:getArcMapLayout aims:map

```
<aims:envelope
    Attributes that set values:
    maxx ="double"
    maxy ="double"
    minx ="double"
    miny ="double"
>

    No Child Elements
</aims:envelope >
```

Bold: Attribute or child element is required.

Description:

Defines the extent of a map.

Restrictions:

None

Notes:

When using aims:envelope as a nested tag in aims:getArcMapLayout or aims:arcMapService, the units are in page units rather than map units. This envelope is referencing an ArcMap layout rather than a map.

Attribute descriptions for envelope:

Attribute	Usage
maxx	Maximum x-coordinate in map or page units.
maxy	Maximum y-coordinate in map or page units.
minx	Minimum x-coordinate in map or page units.
miny	Minimum y-coordinate in map or page units.

Examples for envelope:

Example 1: When using map units.

```
<aims:tcpConnection id="myTCPConnection" host="myMachine"port="5300" debug="true"/>
<aims:mapService id="myService" connectionId="<%=myTCPConnection%>" name="serviceName"/>
<aims:map id="myMapURL" serviceId="<%=myService%>">
  <aims:envelope minx="-120.0" miny="35.0" maxx="-90.0" maxy="60.0" />
</aims:map>
<img src="<%=(myMapURL)%>" />
```

Example 2: When using page units referencing an ArcMap layout.

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>JSP Connector Test</title>
<LINK STYLE="text/css" REL="stylesheet" HREF="font.css">
</head>
<body>
<aims:tcpConnection id="myConnection" host="localhost" port="5300" debug="true" />

<aims:arcMapService id="arcmyDataframeCollection" connectionId="<%=(myConnection)%>"
name="serviceName" error="arcmyError" />
```

```
<aims:getArcMapLayout dataframeCollection="<%=(arcmyDataframeCollection)%>" width="500"
height="700">
  <aims:envelope minx="0" miny="0" maxx="50" maxy="40" />
  <aims:layoutOutput type="jpg" />
  <aims:dataframe dataframeName="South East Asia">
    <aims:featureCoordSys id="102030" />
    <aims:filterCoordSys id="102030" />
    <aims:scale rf="3251279" x="-5079159.83" y="2146309.62"  />
  </aims:dataframe>
</aims:getArcMapLayout>

<aims:getArcMapLayoutAttribute id="arclayoutURL" value="LAYOUTURL"
dataframeCollection="<%=(arcmyDataframeCollection)%>" />
<img src="<%=(arclayoutURL)%>" /><br>

</body>
</html>
```

exact

Used in: mapping
Parent elements: aims:valueMapLabelRenderer aims:valueMapRenderer

```
<aims:exact
    Attributes that set values:
    label ="string"
    method ="IsExact | isContained"  [IsExact]
    value ="string, numeric, or date"
>


    When parent element is aims:valueMapLabelRenderer:
    <calloutMarkerSymbol... />  [Or]
    <chartSymbol... />  [Or]
    <rasterShieldSymbol... />  [Or]
    <shieldSymbol... />  [Or]
    <textSymbol... />  [Or]

    When parent element is aims:valueMapRenderer:
    <gradientFillSymbol... />  [Or]
    <hashLineSymbol... />  [Or]
    <rasterFillSymbol... />  [Or]
    <rasterMarkerSymbol... />  [Or]
    <simpleLineSymbol... />  [Or]
    <simpleMarkerSymbol... />  [Or]
    <simplePolygonSymbol... />  [Or]
    <trueTypeMarkerSymbol... />  [Or]

</aims:exact >
```

Description:

Used with value maps for matching exact values within a specified field in the database.

Restrictions:

Not valid with ArcMap Server.

Notes:

If there are leading or trailing blanks in a field value, they will be trimmed before a comparison is made. For example, a field value of " Hello " is interpreted as "Hello".

Attribute descriptions for exact:

Attribute	Usage
label	Label for legend.
method	Refers to the way a value in the data field is compared to the exact value. Use "isExact" for an exact match. Use "isContained" to search for the value anywhere in a string. String comparisons are case sensitive.
value	Values used for matching records in a selected field. They can be a numeric, string, or date value. Multiple values can be grouped together for one exact category. The default separator between values is a semicolon. To use another separator, use aims:separators.

Examples for exact:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />
<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="350">
<aims:layer layerId="0" visible="true" >
    <aims:valueMapRenderer lookUpField="TRACT">
        <aims:exact label="Tract#5121" value="5121" >
            <aims:simplePolygonSymbol fillColor="255,0,0" fillTransparency="1.0"/>
        </aims:exact>
```

```
        <aims:exact label="Tract#511898" value="511898" >
            <aims:simplePolygonSymbol fillColor="0,255,0" fillTransparency="1.0"/>
        </aims:exact>
        <aims:other label="other">
            <aims:simplePolygonSymbol fillColor="0,0,255" fillTransparency="1.0"/>
        </aims:other>
    </aims:valueMapRenderer>
</aims:layer>
</aims:map>
<img src="<%=(myMap)%>" />
```

extractOutput

Used in: mapping
Parent elements: aims:map

```
<aims:extractOutput
    Attributes that set values:
    baseURL ="string"
    name ="string"
    path ="string"
    url ="string"
>

    No Child Elements
</aims:extractOutput >
```

Description:

Defines a pathname and URL for extracted Zip files.

Restrictions:

ExtractOutput works with paired attributes. If one of the attributes is used, its pair is also required. The attribute pairs are listed in the table below.

Attribute	Paired Attribute	Filename Assignment
path	baseURL	ArcIMS assigns random filename.
name	url	User assigns a filename.

Notes:

If aims:extractOutput is used, the output files are not automatically deleted by ArcIMS Tasker. In order for the files to be deleted, the *taskfile* property must be set in tasker.properties. For information on setting this property, see the ArcIMS Help.

Attribute descriptions for extractOutput:

Attribute	Usage
baseURL	Paired with *path*. URL of output directory if default filename is generated by ArcIMS. Do not include a filename.
name	Paired with *url*. User assigns an output Zip filename. Only filenames with a *.zip extension are valid. Use full pathname along with the filename. The filename must match the filename used in *url*. UNC pathnames are valid (\\myComputer\arcims\output\myfile.zip).
path	Paired with *baseURL*. Directory to output Zip file generated by Extract Server. Do not include the filename. UNC pathnames are valid (\\myComputer\arcims\output).
url	Paired with *name*. URL of output Zip file. Include filename as part of URL. The filename must match the filename used in *name*.

Examples for extractOutput:

Example 1: When using name and url attribute pair.

```
<%@page contentType="text/html"%>
<%@taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>extractOutput tag example</title>
</head>
<body>
<aims:tcpConnection id="myTCPConnection"  host="localhost" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>"
name="SantaClaraExtract" />

<aims:map id="myMap" extract="true" serviceId="<%=(myMapService)%>" width="500"
```

```
height="300">
  <aims:layer layerId="3" extractable="true" />
  <aims:extractOutput name="C:/ArcIMS/Output/TestExtract.zip"
url="http://localhost/output/TestExtract.zip" />
</aims:map>

</body>
</html>
```

Example 2: When using path and baseURL attribute pair.

```
<%@page contentType="text/html"%>
<%@taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>extractOutput tag example</title>
</head>
<body>
<aims:tcpConnection id="myTCPConnection"  host="localhost" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>"
name="SantaClaraExtract" />

<aims:map id="myMap" extract="true" serviceId="<%=(myMapService)%>" width="500"
height="300">
  <aims:layer layerId="3" extractable="true" />
  <aims:extractOutput path="C:/ArcIMS/Output" baseURL="http://localhost/output" />
</aims:map>

</body>
</html>
```

featureCoordSys

Used in: mapping
Parent elements: aims:dataframe aims:getArcMapLayout aims:map

```
<aims:featureCoordSys
    Attributes that set values:
    datumTransformId ="integer"
    datumTransformString ="string"
    id ="integer"
    string ="string"
>

    No Child Elements
</aims:featureCoordSys >
```

Description:

The projection coordinate system to which layers in an ArcIMS service are projected.

Restrictions:

- Must use either *id* or *string*, but not both.
- For datum transformations either *datumTransformId* or *datumTransformString* is used, but not both.
- In ArcMap Image Services, *datumtransformid* or *datumtransformstring* are not valid.

Notes:

- For a complete list of supported IDs and definition strings, see:
    - Projected Coordinate Systems Listing [Sorted by projection ID] [Sorted by name]
    - Geographic Coordinate Systems Listing [Sorted by projection ID] [Sorted by name]

- Datum Transformation Listing [Sorted by projection ID] [Sorted by name]

- When using definition strings, the quotes in the string must be preceded by a backslash (\) so both the Java Connector and the ArcIMS Spatial Server can interpret the string correctly. For example, the definition string for World Mollweide is:

  PROJCS["World_Mollweide",GEOGCS["GCS_WGS_1984",DATUM["D_WGS_1984",SPHEROID["WGS_1984",6378137,298.257223563]],PRIMEM["Greenwich",0],UNIT["Degree",0.0174532925199433295]],PROJECTION["Mollweide"],PARAMETER["False_Easting",0],PARAMETER["False_Northing",0],PARAMETER["Central_Meridian",0],UNIT["Meter",1]]

  Once the string is modified, it should look like this:

  PROJCS[\"World_Mollweide\",GEOGCS[\"GCS_WGS_1984\",DATUM[\"D_WGS_1984\",SPHEROID[\"WGS_1984\",6378137,298.257223563]],PRIMEM[\"Greenwich\",0],UNIT[\"Degree\",0.0174532925199433295]],PROJECTION[\"Mollweide\"],PARAMETER[\"False_Easting\",0],PARAMETER[\"False_Northing\",0],PARAMETER[\"Central_Meridian\",0],UNIT[\"Meter\",1]]

  Note that the syntax for escaping quotes with the Java Connector is different for the Servlet Connector.

- The attributes *datumTransformId* and *datumTransformString* are used when datum transformation information needs to be included. Only datum transformations to and from WGS 1984 are supported.

  - When these attributes are used with aims:coordSys and aims:filterCoordSys, the datum transformation is from a non-WGS 1984 datum to WGS 1984. For example, Pulkovo_1942_To_WGS_1984 (datumtransformid="8157") transforms data from Pulkovo 1942 to WGS 1984.
  - When these attributes are used with aims:featureCoordSys, the datum transformation is from WGS 1984 to a non-WGS 1984 datum. In the above example, the datum transformation is from WGS 1984 to Pulkovo 1942.

- If a layer does not project, double-check that a *.prj file or ArcSDE spatial reference table exists for the layer. If not, aims:coordSys must be included with the layer.

Attribute descriptions for featureCoordSys:

Attribute	Usage
datumTransformId	Datum transformation ID.
datumTransformString	Datum transformation definition string.
id	Projected or geographic coordinate system ID.
string	Projected or geographic coordinate system definition string.

Examples for featureCoordSys:

Example 1:

```
<aims:tcpConnection id="myTCPConnection"  host="myMachine" port="5300" />
<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>"
      name="<%=myService%>" />
<aims:map id="myMap" legend="myLegend" envelope="myEnvelope"
serviceId="<%=(myMapService)%>" width="300" height="100" background="100,140,230"
transcolor="0,0,0" bufferImage="true" bufferRegion="true">
  <aims:featureCoordSys id="54008"/>
  <aims:filterCoordSys id="4326" />
</aims:map>
```

filter

Used in: mapping
Parent elements: aims:createLayer aims:layer

```
<aims:filter
    Attributes that set values:
    accuracy ="Distance between points"  [0]
    checkESC ="true | false"  [false]
    envelope ="true | false"  [false]
    featureLimit ="integer"  [All that meet criteria]
    geometry ="true | false"  [true]
    globalEnvelope ="true | false"  [false]
    joinExpression ="string"
    joinTables ="string"
    relation ="area_intersection | envelope_intersection"  [area_intersection]
    searchOrder ="optimize | spatialfirst | attributefirst"  [optimize]
    subFields ="#ALL# | #ID# | #SHAPE# | Other fields in database"  [#ALL#]
    whereExpression ="string"
>

    <buffer... />
    <envelope... />
    <line... />
    <point... />
    <polygon... />

</aims:filter >
```

Description:

Sets a filter on a layer by querying dataset attributes or by making a spatial query.

Restrictions:

- The attributes *accuracy*, *joinExpression*, and *joinTables* are not valid with ArcMap Image Services.
- When joining shapefiles, shapefile layers can only be joined to other DBF files located in the same directory as the shapefile. A joined DBF file cannot be another shapefile DBF file that is currently being used in an ArcIMS service.
- DBF jointable names are limited to 10 characters.
- When joining tables in ArcSDE, ArcSDE layers can only be joined to other tables in the relational database management system (RDBMS).

Notes:

- Queries defined in a map configuration file always filter the data and cannot be changed through a request. Any requests made to a filtered layer in a map configuration file can select only features within the filtered subset. For example, assume a map configuration file has a world cities layer that has been filtered to display only cities with a population greater than 1,000,000. All requests to this layer, whether tabular or spatial, include only cities with a population greater than 1,000,000. Other cities in the database are ignored.
- When using aims:filter, one of the options is to return the geometry of selected features by setting the *geometry* attribute to "true". To return geometry, the #SHAPE# field (or #ALL#) must be included in the *subFields* list.
- When creating a *where* clause, two fields from the same attribute table can be used for comparison. For example, a query can be made to find states with a female population greater than the male population.

  <aims:filter whereExpression="COUNTRY.STATE.FEMALES > COUNTRY.STATE.MALES" />
- Some symbols must be "escaped" inside a *where* expression:
  ampersand (&) is escaped to &
  double quotes (") are escaped to "
  single quotes (') are escaped to '
  greater than (>) is escaped to >
  less than (<) is escaped to <
- **Joining ArcSDE tables.** When joining ArcSDE tables, all valid joins between the RDBMS and ArcSDE are also valid in ArcIMS. To join ArcSDE tables, a *where* clause is used for defining which tables are joined and for any additional filters using SQL syntax. Multiple joins are also permitted.

  In the following example, the *where* statement includes setting up two relates (in bold type) and filtering the selection to FIPS_CNTRY='CA' (in italic type).

<aims:filter whereExpression=**"DB.CITY.CITY_FIPS = DB.SCHOOLS.CITY_FIPS and DB.SCHOOLS.SCHOOL_ID = DB.SCHOOL_STATS.SCHOOL_ID** and *DB.CITY.CNTY_FIPS='013'"* jointables="DB.SCHOOLS DB.SCHOOL_INFO" />

In this statement, cities (DB.CITY) are first joined to schools (DB.SCHOOLS) using the column CITY_FIPS as the common join item. Next, the schools (DB.SCHOOLS) are joined to a table with statistics on the schools (DB.SCHOOL_STATS), using SCHOOL_ID as the join item. A further filter is placed on this query by limiting the results to schools located within a specified county, in this example, where CNTY_FIPS='013'.

The list of joined tables must be under the *jointables* attribute. If more than one table is joined, table names are separated by a space. When naming tables, the full ArcSDE table name must be used. In the above example, the two joined tables are DB.SCHOOLS and DB.SCHOOL_INFO.

- **Joining DBF files.** When joining DBF tables, joins can be made between the shapefile DBF table and one or more external DBF tables that reside in the same directory as the shapefile. One restriction is that a joined DBF file cannot be another shapefile DBF file that is currently being used in a service.

  To join DBF files, the *joinexpression* attribute is used. This attribute uses the syntax: "To=[master table column which will be used for joining], From=[defines a join table column which will be joined], Type=[exact or scan]".

To	Refers to the master DBF table and defines the field that is used for joining. When referring to this table, the DBF table name must be used as a prefix to the field name. The entire expression is surrounded by square brackets, for example, joinexpression="To=[mastertable.fieldname]".
From	Refers to the DBF table that is joined to the master DBF table and the field that is used for joining, for example, joinexpression="From=[jointable.fieldname]".
Type=[exact]	Defines an exact match relation that permits only a single match between the master and join tables. Both one-to-one and many-to-one relations are exact match relations. In a one-to-one relation, there is only one record in the master that matches a single record in the join table. In a many-to-one relation, there are one or more records in the master that match a single join record. If there are multiple join records that match a single master record, then a composite record is only generated for the first join record.
Type=[scan]	In a scan relation, if there are multiple join records for a master record, there is one composite record in the extended data file for each of the matching join records. Both one-to-many and many-to-many relations are scan relations. In a one-to-many relation, each record in the master can have multiple matching join records. A many-to-many relation is the same as one-to-many, except that different master records can match the same join record.

- To, From, and Type parameters are case sensitive (first letter is capitalized), and they are separated by a comma, ",". Any number of joined tables can be defined in the *joinExpression* attribute. Joined tables are separated by a semicolon, ";", such as the following:

  joinExpression="To=[A.ID],From=[B.ID],Type=[scan];To=[B.NAME],From=[C.NAME], Type=[exact] "

  All joined DBF tables must be listed under *joinTables*. If multiple tables are joined, table names are separated by a space. When naming tables, the name of the DBF file without the extension is used. An example DBF query expression is:

  <aims:filter joinExpression="To=[counties.CNTY_FIPS],From=[countyinfo.FIPS],Type=[scan]; To=[countyinfo.FIPS],From=[state_roads.FIPS],Type=[scan]" joinTables="countyinfo state_roads" whereExpression="counties. NAME='Washoe'" />

  In this example, a county DBF file (counties.dbf) is first joined to a DBF file containing county information (countyinfo.dbf). They are joined on the field FIPS (counties.CNTY_FIPS and countyinfo.FIPS). Next, countyinfo.dbf is joined to state_roads.dbf, once again using FIPS as the join item. The jointables are listed as countyinfo and state_roads, separated by a space. The query is filtered to only include one county, in this case Washoe.
- When joining DBF files to a shapefile, the DBF file cannot be read-only.
- **Querying with dates.** The syntax for querying dates is the same regardless of the locale. A date query uses the following syntax:

  {ts 'YYYY-MM-DD hh:mi:ss'}

  where

YYYY	Year	Required	Use four digits for the year.
MM	Month (01-12)	Required	Use two digits for the month. March is 03.
DD	Day (01-31)	Required	Use two digits for the day. The fourth is 04.
hh	Hour (00-23)	Optional	Use a 24-hour clock. 8 a.m. is 08, and 8 p.m. is 20.
mi	Minutes (00-59)	Optional	Use two digits for the minutes. If minutes is used, hours must also be included.
ss	Seconds (00-59)	Optional	Use two digits for the seconds. If seconds is used, hours and minutes must also be included.

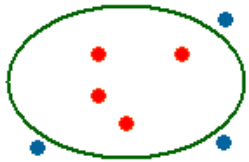
- The year, month, and day are each separated by a dash (-). The hour, minutes, and seconds are each separated by a colon (:). The date is enclosed in single quotes (') inside curly brackets ({}). Before the date, ts (for time stamp) must be included.

  For 8:03:23 a.m. January 4, 2000, the query on a DBF file looks like:
  <aims:filter whereExpression="MYDATE = {ts '2000-01-04 08:03:23'}" />

  For 9:18 p.m. March 8, 2002, the query on an ArcSDE layer looks like:
  <aims:filter whereExpression="ARCSDE.TABLE.MYDATE = {ts '2002-03-08 21:18:00'}" />

- When "area_intersection" is used with *relation*, all features that partially or fully fall within the area of the filter are selected. This results in a complex operation that can take some time. To speed up processing, "envelope_intersection" can be used instead. This method checks to see if the bounding box of the filter overlaps any bounding boxes of the features in the layer. A much quicker search results, but the features found could fall outside the area of the filter. In the examples below, an oval shape is used as the filter. When "envelope_intersection" is used, features outside the filter can be selected, even though they are far outside the filter.

Shape of spatial filter to select points

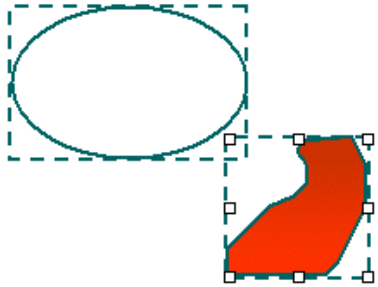


Points selected when relation="area_intersection"

Points selected when relation="envelope_intersection"

Shape of spatial filter to select polygons

No polygons are selected when relation="area_intersection"



Polygon is selected when relation="envelope_intersection"

Attribute descriptions for filter:

Attribute	Usage
accuracy	Generalizes geometry when *geometry="true"* based on the distance specified and the resolution of the image. Units are the same as the service.
checkESC	Used to determine if the returned data should include escaped characters for ampersand, single quote, double quote, less than, and greater than. For example, if the value for SUB_REGION is "P&NW", then: • If checkesc="false", in the response the value of SUB_REGION is "P&NW". • If checkesc="true", the value of SUB_REGION is "P&NW".
envelope	Used to request the bounding envelope of each returned feature.
featureLimit	Maximum number of features to be extracted that meet criteria.
geometry	Requests feature coordinates.
globalEnvelope	When set to "true", the overall envelope for all the features returned is included in the response. Because the number of features returned depends on *beginrecord* and *featurelimit*, only the actual records extracted based on *featurelimit* are included in the overall envelope. Note that in order for the global envelope to be returned, *subfields* must include either #SHAPE# or #ALL#.
joinExpression	Used for joined tables with DBF files only; *jointables* must be filled to contain list of tables used; not required when a table join is done on ArcSDE. String must form expression: "To=[master table column which will be used for joining], From=[defines a join table column which will be joined], Type=[exact or scan]".
joinTables	List of joined table names separated by blank spaces; for ArcSDE, table name is full name including database name—for example, DATA.STATE; for shapefiles, use the DBF filename without the extension—for example, STATES.
relation	Describes spatial relation. See Notes section for more details.
searchOrder	Used with ArcSDE layers only. Determines whether the attribute or spatial part of an ArcSDE query is processed first. "Spatialfirst" processes the spatial part of the query before the attribute part. "Attributefirst" processes the attribute part of the query first. If "optimize" is used, ArcSDE makes the judgment whether to pick "spatialfirst" or "attributefirst".

subFields	List of fields available for querying or extracting. Multiple fields can be included in the *subfields* list. Fields must be separated by a blank space. If *subfields* is not used, all fields are returned. If *subfields* is used, only listed fields are returned. The subfields #SHAPE# or #ALL# must be included if geometry is to be returned. In addition, the *geometry* attribute must be set to "true". The *subfields* list can include fields from the layer table or a joined table. • For shapefiles with no joined tables, the field can be referenced using the short format. field="AREA" • For shapefiles with joined tables, the name of the joined table must be included along with the field. field="JOINEDTABLE.AREA" • For ArcSDE layers with or without joined tables, the field must be referenced using the full long format. field="ARCSDENAME.TABLE.AREA"
whereExpression	Defines *where* part of SQL expression. Required when *jointables* attribute for ArcSDE tables is used. See Notes section for information on querying dates.

Examples for filter:

Example 1: When using a buffer as a filter.

```
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300"
bufferImage="true"
bufferRegion="true">
  <aims:layer layerId="3" visible="true">
    <aims:filter whereExpression="NAME='COYOTE RIVER'">
      <aims:buffer distance="5" bufferUnits="Miles" performBuffer="true">
        <aims:bufferLayer>
          <aims:simplePolygonSymbol transparency="0.5" fillColor="2,255,250"
boundaryColor="255,255,255" />
        </aims:bufferLayer>
```

```
        <aims:targetLayer targetLayerId="5">
          <aims:simpleMarkerSymbol color="0,0,0" type="star" width="20" />
        </aims:targetLayer>
      </aims:buffer>
    </aims:filter>
  </aims:layer>
</aims:map>
<img src="<%=myMap%>" >
```

Example 2:

```
<aims:tcpConnection id="myTCPConnection"  host="myMachine" port="5300" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>"
name="<%=myService%>" loadRecordset="true" loadEnvelope="true" loadExtensions="true"
loadRenderer="true" />

<aims:map id="myMap" legend="myLegend" envelope="myEnvelope"
serviceId="<%=myMapService%>" width="500" height="300">
  <aims:separators  ts="," cs=" " />
  <aims:layer layerId="0" visible="true">
    <aims:filter envelope="true" geometry="true" globalEnvelope="true">
      <aims:line coords="-122.483 37.769,-122.482 37.763" />
    </aims:filter>
    <aims:recordset>
      <aims:iterateTableDesc fieldName="fName" fieldType="fType" fieldPrecision="fPrec"
fieldLength="fLen">
        <%=(fName)%>
        <%=(fType)%>
        <%=(fPrec)%>
        <%=(fLen)%> <br>
      </aims:iterateTableDesc>
    </aims:recordset>
  </aims:layer>
</aims:map>
```

filterCoordSys

Used in: mapping
Parent elements: aims:dataframe aims:getArcMapLayout aims:map

```
<aims:filterCoordSys
    Attributes that set values:
    datumTransformId ="integer"
    datumTransformString ="string"
    id ="integer"
    string ="string"
>

    No Child Elements
</aims:filterCoordSys >
```

Description:

The current coordinate system of the requesting client.

Restrictions:

- Must use either *id* or *string* but not both.
- For datum transformations either *datumTransformId* or *datumTransformString* is used but not both.
- In ArcMap Image Services, *datumtransformid* and *datumtransformstring* are not valid.

Notes:

- For a complete list of supported IDs and definition strings, see:
  - Projected Coordinate Systems Listing [Sorted by projection ID] [Sorted by name]
  - Geographic Coordinate Systems Listing [Sorted by projection ID] [Sorted by name]

    - Datum Transformation Listing [Sorted by projection ID] [Sorted by name]

- When using definition strings, the quotes in the string must be preceded by a backslash (\) so both the Java Connector and the ArcIMS Spatial Server can interpret the string correctly. For example, the definition string for World Mollweide is:

  PROJCS["World_Mollweide",GEOGCS["GCS_WGS_1984",DATUM["D_WGS_1984",SPHEROID["WGS_1984",6378137,298.257223563]],PRIMEM["Greenwich",0],UNIT["Degree",0.0174532925199433295]],PROJECTION["Mollweide"],PARAMETER["False_Easting",0],PARAMETER["False_Northing",0],PARAMETER["Central_Meridian",0],UNIT["Meter",1]]

  Once the string is modified, it should look like this:

  PROJCS[\"World_Mollweide\",GEOGCS[\"GCS_WGS_1984\",DATUM[\"D_WGS_1984\",SPHEROID[\"WGS_1984\",6378137,298.257223563]],PRIMEM[\"Greenwich\",0],UNIT[\"Degree\",0.0174532925199433295]],PROJECTION[\"Mollweide\"],PARAMETER[\"False_Easting\",0],PARAMETER[\"False_Northing\",0],PARAMETER[\"Central_Meridian\",0],UNIT[\"Meter\",1]]

  Note that the syntax for escaping quotes with the Java Connector is different for the Servlet Connector.

- The attributes *datumTransformId* and *datumTransformString* are used when datum transformation information needs to be included. Only datum transformations to and from WGS 1984 are supported.
    - When these attributes are used with aims:coordSys and aims:filterCoordSys, the datum transformation is from a non-WGS 1984 datum to WGS 1984. For example, Pulkovo_1942_To_WGS_1984 (datumtransformid="8157") transforms data from Pulkovo 1942 to WGS 1984.
    - When these attributes are used with aims:featureCoordSys, the datum transformation is from WGS 1984 to a non-WGS 1984 datum. In the above example, the datum transformation is from WGS 1984 to Pulkovo 1942.

- If a layer does not project, double-check that a *.prj file or ArcSDE spatial reference table exists for the layer. If not, aims:coordSys must be included with the layer.

Attribute descriptions for filterCoordSys:

Attribute	Usage
datumTransformId	Datum transformation ID.
datumTransformString	Datum transformation definition string.
id	Projected or geographic coordinate system ID.
string	Projected or geographic coordinate system definition string.

Examples for filterCoordSys:

Example 1:

```
<aims:tcpConnection id="myTCPConnection"  host="myMachine" port="5300" />
<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>"
      name="<%=myService%>" />
<aims:map id="myMap" legend="myLegend" envelope="myEnvelope"
serviceId="<%=(myMapService)%>" width="300" height="100" background="100,140,230"
transcolor="0,0,0" bufferImage="true" bufferRegion="true">
  <aims:featureCoordSys id="54008"/>
  <aims:filterCoordSys id="4326" />
</aims:map>
```

getArcMapLayout

Used in: mapping
Parent elements: None

```
<aims:getArcMapLayout

    Attributes that set values:
    autoresize ="true | false"  [false]
    dpi ="1 - NNN"
    height ="integer"
    width ="integer"

    Attributes that pass objects:
    dataframeCollection ="object"
>

    <dataframe... />
    <envelope... />
    <featureCoordSys... />
    <filterCoordSys... />
    <iterateDataframe... />
    <layoutOutput... />

</aims:getArcMapLayout >
```

Bold: Attribute or child element is required.

Description:

Main tag for requesting an ArcMap layout.

Restrictions:

When using *dpi*, you must also include *height* and *width*.

Notes:

When using aims:envelope as a nested tag, the units are in page units rather than map units. This envelope is referencing an ArcMap layout rather than a map.

Attribute descriptions for getArcMapLayout:

Attribute	Usage
autoresize	The maximum generated image size is based on the image memory limit set when an ArcMap Image Service is started. For example, an image memory limit of 1 MB allows a map no larger than 262,144 pixels (512 x 512) to be generated. If *autoresize* is set to "true", a requested map greater than the maximum pixel count will be reduced in size to within the maximum pixel count. If *autoresize* is set to "false", no image is generated and an error message is returned by the ArcIMS Spatial Server.
dataframeCollection	Data Frames Collection object passed from aims:arcMapService *id*.
dpi	Dots per inch (dpi).
height	Height of the layout.
width	Width of the layout.

Examples for getArcMapLayout:

Example 1:

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>JSP Connector Test</title>
<LINK STYLE="text/css" REL="stylesheet" HREF="font.css">
</head>
<body>
<aims:tcpConnection id="myConnection" host="localhost" port="5300" debug="true" />

<aims:arcMapService id="arcmyDataframeCollection" connectionId="<%=(myConnection)%>"
name="serviceName" error="arcmyError" />
```

```
<aims:getArcMapLayout dataframeCollection="<%=(arcmyDataframeCollection)%>" width="500"
height="700">
  <aims:envelope minx="0" miny="0" maxx="50" maxy="40" />
  <aims:layoutOutput type="jpg" />
  <aims:dataframe dataframeName="South East Asia">
    <aims:featureCoordSys id="102030" />
    <aims:filterCoordSys id="102030" />
    <aims:scale rf="3251279" x="-5079159.83" y="2146309.62"  />
  </aims:dataframe>
</aims:getArcMapLayout>

<aims:getArcMapLayoutAttribute id="arclayoutURL" value="LAYOUTURL"
dataframeCollection="<%=(arcmyDataframeCollection)%>" />
<img src="<%=(arclayoutURL)%>" /><br>

</body>
</html>
```

getArcMapLayoutAttribute

Used in: **mapping**
Parent elements: None

<aims:getArcMapLayoutAttribute

Attributes that set values:
value ="IMAGEHEIGHT | IMAGEWIDTH | OUTPUTTYPE | PAGEUNITS | MINX | MINY | MAXX | MAXY | FEATUREID | FEATURESTRING | FILTERID | FILTERSTRING | LAYOUTPATH | LAYOUTTYPE | LAYOUTURL"

Attributes that pass objects:
dataframeCollection ="*object*"

Attributes that return values:
id ="*string*"
>

No Child Elements
</aims:getArcMapLayoutAttribute >

Bold: Attribute or child element is required.

Description:

Returns a layout property.

Restrictions:

None

Notes:

None

Attribute descriptions for getArcMapLayoutAttribute:

Attribute	Usage
dataframeCollection	Data Frames Collection object passed from aims:arcMapService *id*.
id	Returns the value of the property specified in *value*.
value	Name of the layout property to retrieve.

Examples for getArcMapLayoutAttribute:

Example 1:

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>JSP Connector Test</title>
<LINK STYLE="text/css" REL="stylesheet" HREF="font.css">
</head>
<body>
<aims:tcpConnection id="myConnection" host="localhost" port="5300" debug="true" />

<aims:arcMapService id="arcmyDataframeCollection" connectionId="<%=(myConnection)%>"
name="serviceName" error="arcmyError" />

<aims:getArcMapLayout dataframeCollection="<%=(arcmyDataframeCollection)%>"/>

<aims:getArcMapLayoutAttribute id="url" value="LAYOUTURL"
dataframeCollection="<%=(arcmyDataframeCollection)%>" />
<img src="<%=(url)%>" /><br>

<aims:getArcMapLayoutAttribute id="pageunits" value="PAGEUNITS"
dataframeCollection="<%=(arcmyDataframeCollection)%>" />
Page Units : <%=(pageunits)%> <br>

</body>
</html>
```

getContainerAttribute

Used in: administration
Parent elements: aims:iterateContainersForServer

```
<aims:getContainerAttribute
    Attributes that set values:
    attribute ="NAME | RUNNING_THREADS | INSTANCES_ALLOWED"
    virtualServerType ="ImageServer | FeatureServer | MetadataServer | QueryServer | ExtractServer | GeocodeServer"
    virtualServerVersion ="" | ArcMap"

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    id ="string"
    virtualServers ="object"
>

    No Child Elements
</aims:getContainerAttribute >
```

Bold: Attribute or child element is required.

Description:

Retrieves information about an ArcIMS Spatial Server and all of the Virtual Servers associated with this Spatial Server.

Restrictions:

None

Notes:

None

Attribute descriptions for getContainerAttribute:

Attribute	Usage
attribute	The name of the attribute to retrieve about an ArcIMS Spatial Server. If the attribute is not recognized, an empty string is returned.
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection. Required only when *virtualServers* is used.
id	Returns a string containing the value of the attribute specified.
virtualServers	Returns a Virtual Server collection. Use aims:iterateVirtualServers to list the values of each Virtual Server.
virtualServerType	The Virtual Server type: ImageServer, FeatureServer, MetadataServer, ExtractServer, GeocodeServer, QueryServer. For ArcMap Server, use "ImageServer" for this attribute.
virtualServerVersion	Specifies whether an ImageServer is an ArcMap Image Server. For ArcMap Image Servers, use "ArcMap". For all other server types, use "".

Examples for getContainerAttribute:

Example 1:

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://mymachine"
      role="admin"
      debug="true"
      username="admin"
      password="admin"
/>

<aims:getServer id="myServer" connectionId="<%=myhttpConnection%>" name="mymachine"
error="myError" />

<aims:iterateContainersForServer server="<%=myServer%>" count="containerCount" >
      <aims:getContainerAttribute id="myContainer" attribute="NAME"
connectionId="<%=myhttpConnection%>" virtualServers="myVirtualServers" />
```

```
      <aims:getContainerAttribute id="arcmapthreads" attribute="RUNNING_THREADS"
virtualServerType="ImageServer" virtualServerVersion="ArcMap" />
      <aims:getContainerAttribute id="arcmapallowed" attribute="INSTANCES_ALLOWED"
virtualServerType="ImageServer" virtualServerVersion="ArcMap" />
      <td>
        <br><b>Container Name is: <%=myContainer%></b>
        <br><b>Running Threads of ImageServer ArcMap: <%=arcmapthreads%></b>
        <br><b>Allowed to add ImageServer ArcMap type to container:
<%=arcmapallowed%></b>
      </td>
      <aims:iterateVirtualServers virtualServers="<%=myVirtualServers%>"
count="serviceCount2" >
            <aims:getVirtualServerAttribute id="name" attribute="NAME" />
            <aims:getVirtualServerAttribute id="type" attribute="TYPE" />
            <aims:getVirtualServerAttribute id="access" attribute="ACCESS" />
            <aims:getVirtualServerAttribute id="description" attribute="DESCRIPTION" />
            <aims:getVirtualServerAttribute id="version" attribute="VERSION" />
            <aims:getVirtualServerAttribute id="containerset" attribute="CONTAINERSET" />
            <td>
              <br><b>name is: <%=name%></b>
              type is: <%=type%>
              description is: <%=description%>
              version is: <%=version%>
              access is: <%=access%>
              containerset is: <%=containerset%>
            </td>
      </aims:iterateVirtualServers>

</aims:iterateContainersForServer>
containerCount <%=containerCount%>
```

getContentInfo

Used in: metadata
Parent elements: None

```
<aims:getContentInfo
    Attributes that set values:
    service ="string"

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    id ="string"
>

    No Child Elements
</aims:getContentInfo >
```

Bold: Attribute or child element is required.

Description:

Requests whether content is validated by clients.

Restrictions:

None

Notes:

None

Attribute descriptions for getContentInfo:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
id	Returns true or false depending on whether content is valid or not.
service	The name of service.

Examples for getContentInfo:

Example 1:

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
  <title>Tag "getContentInfo" testing</title>
</head>
<body>
<aims:tcpConnection id="myConnection" host="machineName" port="5300" debug="true" />
<aims:getContentInfo id="validate" connectionId="<%=(myConnection)%>"
service="metadataServiceName"/>
<%
  System.out.println("GetContentInfo Validate : " + validate);
%>
</body>
</html>
```

getElement

Used in: metadata
Parent elements: None

```
<aims:getElement

    Attributes that pass objects:
    loadedElements ="object"

    Attributes that return values:
    id ="string"
    value ="string"
>

    No Child Elements
</aims:getElement >
```

Bold: Attribute or child element is required.

Description:

Retrieves the value of a metadata element.

Restrictions:

The element value must be loaded into memory using aims:loadElements before it can be accessed using aims:getElement.

Notes:

None

Attribute descriptions for getElement:

Attribute	Usage
id	Returns the element's text value.
loadedElements	The set of elements retrieved using aims:loadElements.
value	Defines the xpath or location of an element inside an XML tree. For example, the element "title" might be inside an XML structure such as <idinfo> <citation> <citeinfo> **<title>...</title>** </citeinfo> </citation> </idinfo> To define "title", the string value for this attribute would be "idinfo/citation/citeinfo/**title**".

Examples for getElement:

Example 1: Retrieves the value of a metadata element.

```
<%
String FGDCElements = "metadata/idinfo/citation/citeinfo/title,"
      + "metadata/idinfo/keywords/place/placekey,"
      + "metadata/spdoinfo/direct,"
      + "metadata/idinfo/browse/browsen,"
      + "metadata/idinfo/browse/browsed,"
      + "metadata/idinfo/citation/citeinfo/pubinfo/publish,"
      + "metadata/dataqual/lineage/srcinfo/srcscale,"
      + "metadata/Esri/qn/resolution,";
%>

<%-- Load all the element values we'll be using --%>
```

```
<aims:loadElements id="theElements" url="<%= metadataUrl %>" names="<%= FGDCElements %>"
/>

<%-- Get the dataset's title --%>
<aims:getElement id="theTitle" loadedElements="<%= theElements %>"
value="metadata/idinfo/citation/citeinfo/title" />

<%-- Get the dataset's publisher --%>
<aims:getElement id="publish" loadedElements="<%= theElements %>"
value="metadata/idinfo/citation/citeinfo/pubinfo/publish" />

<P>
<b>Title:</b> <%= theTitle %><br>
<P>
<b>Publisher:</b> <%= publish %><br>
```

getEnvelope

Used in: mapping
Parent elements: aims:layer aims:map

```
<aims:getEnvelope
    Attributes that pass objects:
    serviceId ="object"

    Attributes that return values:
    maxx ="double"
    maxy ="double"
    minx ="double"
    miny ="double"
>

    No Child Elements
</aims:getEnvelope >
```

Bold: Attribute or child element is required.

Description:

Retrieves the full extent envelope for a service, the layer extent, or the current extent of a map.

Restrictions:

None

Notes:

When retrieving the envelope of a service, getEnvelope is not nested inside another tag. However, it must be used after aims:mapService is declared.

Attribute descriptions for getEnvelope:

Attribute	Usage
maxx	Maximum x-coordinate in map or page units.
maxy	Maximum y-coordinate in map or page units.
minx	Minimum x-coordinate in map or page units.
miny	Minimum y-coordinate in map or page units.
serviceId	The name of the ArcIMS service. This attribute is needed only when the full extent of the service is requested. It is not needed when requesting map or layer extents.

Examples for getEnvelope:

Example 1: When retrieving the full extent of a service.

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
  <title>getEnvelope outside the map tag</title>
</head>
<body>
<aims:tcpConnection id="myTCPConnection" host="localhost" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="SantaClara"
loadEnvelope="true" />

<aims:getEnvelope  serviceId="<%=(myMapService)%>" minx="minx" miny="miny" maxx="maxx"
maxy="maxy" />
  Min X : <%=(minx)%><br>
  Min Y : <%=(miny)%><br>
  Max X : <%=(maxx)%><br>
  Max Y : <%=(maxy)%><br>

</body>
</html>
```

Example 2: When nested inside aims:map. Retrieves the current extent of the map.

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
  <title>getEnvelope nested inside map tag</title>
</head>
<body>
<aims:tcpConnection id="myTCPConnection" host="localhost" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="SantaClara"
loadEnvelope="true" />

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:getEnvelope  minx="minx" miny="miny" maxx="maxx" maxy="maxy" />
    Min X : <%=(minx)%><br>
    Min Y : <%=(miny)%><br>
    Max X : <%=(maxx)%><br>
    Max Y : <%=(maxy)%><br>
</aims:map>
</body>
</html>
```

Example 3: When nested inside aims:layer. Retrieves the extent of the layer.

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
  <title>getEnvelope nested inside layer tag</title>
</head>
<body>
<aims:tcpConnection id="myTCPConnection" host="localhost" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="SantaClara"
loadEnvelope="true" />

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
```

```
  <aims:layer layerId="5" visible="true">
    <aims:getEnvelope  minx="minx" miny="miny" maxx="maxx" maxy="maxy" />
      Min X : <%=(minx)%><br>
      Min Y : <%=(miny)%><br>
      Max X : <%=(maxx)%><br>
      Max Y : <%=(maxy)%><br>
  </aims:layer>
</aims:map>

</body>
</html>
```

getInputAttribute

Used in: mapping
Parent elements: aims:iterateAddressMatchInputs

```
<aims:getInputAttribute
    Attributes that set values:
    attribute ="DESCRIPTION | LABEL | ID | TYPE | WIDTH"

    Attributes that return values:
    id ="string"
>

    No Child Elements
</aims:getInputAttribute >
```

Bold: Attribute or child element is required.

Description:

Used to retrieve values stored in the addressMatchInputs object.

Restrictions:

None

Notes:

None

Attribute descriptions for getInputAttribute:

Attribute	Usage
attribute	The name of the attribute to return in *id*.
id	Returns the value of the property specified in *attribute*.

Examples for getInputAttribute:

Example 1:

```
<aims:tcpConnection id="mytcpConnection" port="5300" host="brugge" debug="true" />

<aims:mapService id="myService" connectionId="<%=mytcpConnection%>" name="geostreets"
loadExtensions="true" />

<aims:map id="mapImage" serviceId="<%=myService%>" reset="false" >
  <aims:layer layerId="0" >
    <aims:addressMatchInputs id="myInputs" minScore="0" maxCandidates="100" >
      <aims:setInput inputId="STREET" value="380 New York St" />
      <aims:setInput inputId="ZONE" value="" />
      <aims:setInput inputId="CROSSSTREET" value="" />
    </aims:addressMatchInputs>
        <table>
    <aims:iterateAddressMatchInputs inputs="<%=myInputs%>" count="myCount"
minScore="myMinScore" maxCandidates="myMax" style="myStyle" >
      <aims:getInputAttribute id="myDes" attribute="DESCRIPTION" />
      <aims:getInputAttribute id="myId" attribute="ID" />
      <aims:getInputAttribute id="myLabel" attribute="LABEL" />
      <aims:getInputAttribute id="myType" attribute="TYPE" />
      <aims:getInputAttribute id="myWidth" attribute="WIDTH" />
            <tr><td>
        <b>desc is: <%=myDes%></b><br>
           id is: <%=myId%><br>
           label is: <%=myLabel%><br>
           type is: <%=myType%><br>
```

```
            width is: <%=myWidth%>
             </td></tr>
    </aims:iterateAddressMatchInputs>

       <tr><td>
      <b>INPUT count is: <%=myCount%></b><br>
         minScore is: <%=myMinScore%><br>
         maxCandidates is: <%=myMax%><br>
         style is: <%=myStyle%>
    </td></tr>

    <aims:iterateAddressMatchResults inputs="<%=myInputs%>" count="resultsCount" >
      <aims:getResultAttribute id="myValue" attribute="VALUE" />
      <aims:getResultAttribute id="myScore" attribute="SCORE" />
      <aims:getResultAttribute id="myPoint" attribute="POINT" />
      <tr><td>
        <b>value is: <%=myValue%></b><br>
           score is: <%=myScore%><br>
           point is: <%=myPoint%>
      </td></tr>
    </aims:iterateAddressMatchResults>

        <tr><td>
      <br><b>RESULT count is: <%=resultsCount%></b>
        </td></tr></table>
  </aims:layer>
</aims:map>
```

getLayerAttribute

Used in: mapping
Parent elements: aims:iterateLayers

```
<aims:getLayerAttribute

    Attributes that set values:
    attribute ="NAME | ID | VISIBLE | MAXSCALE | MINSCALE | TYPE | FEATURETYPE"

    Attributes that return values:
    id ="string"
>

    No Child Elements
</aims:getLayerAttribute >
```

Bold: Attribute or child element is required.

Description:

Returns layer information.

Restrictions:

None

Notes:

None

Attribute descriptions for getLayerAttribute:

Attribute	Usage
attribute	The layer property to retrieve.
id	Returns the value of the property specified in *attribute*.

Examples for getLayerAttribute:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />
<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="serviceName"
/>
<aims:map id="myMap" serviceId="<%=myMapService%>" width="300" height="100">
<aims:iterateLayers count="layerCount">
  <aims:getLayerAttribute id="layerId" attribute="ID" />
    ID : <%=layerId%>
  <aims:getLayerAttribute id="layerName" attribute="NAME" />
    Layer Name : <%=layerName%><br>
</aims:iterateLayers>
Layer Count : <%=(layerCount)%>
</aims:map>
```

getLayerId

Used in: mapping
Parent elements: aims:map

```
<aims:getLayerId
    Attributes that set values:
    name ="string"

    Attributes that return values:
    id ="string"
>

    No Child Elements
</aims:getLayerId >
```

Bold: Attribute or child element is required.

Description:

Returns the layer ID for a specified layer name.

Restrictions:

If multiple layers in the service have the same name, only the *id* of the first layer in the list is returned.

Notes:

None

Attribute descriptions for getLayerId:

Attribute	Usage
id	Returns the layer ID based on the layer name.
name	Name of the layer.

Examples for getLayerId:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>"
name="<%=myService%>" />

<aims:map id="myMap" legend="myLegend" envelope="myEnvelope"
serviceId="<%=myMapService%>" width="300" height="100" background="100,140,230"
transcolor="0,0,0" bufferImage="true" bufferRegion="true">

  <aims:getLayerId id="myId" name="continents" />
  <%=myId%>

</aims:map>
```

getLayout

Used in: **mapping**
Parent elements: None

```
<aims:getLayout

    Attributes that set values:
    value ="IMAGEHEIGHT | IMAGEWIDTH | OUTPUTTYPE | PAGEUNITS | MINX | MINY | MAXX | MAXY |
FEATUREID | FEATURESTRING | FILTERID | FILTERSTRING | LAYOUTPATH | LAYOUTTYPE | LAYOUTURL"

    Attributes that pass objects:
    dataframeCollection ="object"

    Attributes that return values:
    id ="string"
>

    No Child Elements
</aims:getLayout >
```

Bold: Attribute or child element is required.

Description:

Requests an ArcMap layout. This tag has been deprecated in ArcIMS 4.0.1. Use aims:getArcMapLayoutAttribute instead.

Restrictions:

- Valid only with ArcMap Server.
- **Known limitation with the Java Connector and JSP Tag Library.** When using aims:getLayout, only the default layout as specified in the map document can be retrieved. You can zoom and pan the layout page using aims:envelope as a nested tag in aims:arcMapService. However, you cannot change the projection or extent of individual data frames. When using the Java Connector beans, one projection and extent can be used for all data frames. In order to have complete control of projections and extents in individual data frames, ArcXML must be used.

Notes:

None

Attribute descriptions for getLayout:

Attribute	Usage
dataframeCollection	Data Frames Collection object passed from aims:arcMapService *id*.
id	Returns the value of the property specified in *value*.
value	Name of the layout property to retrieve.

Examples for getLayout:

Example 1: .

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />

<aims:arcMapService id="myDataframeCollection" connectionId="<%=(myConnection)%>"
name="multidf" dpi="0" loadLayout="true" error="myError">
      <aims:envelope minx="10" miny="10" maxx="15" maxy="15" />
</aims:arcMapService>

<aims:getLayout id="layoutURL" value="LAYOUTURL"
dataframeCollection="<%=(myDataframeCollection)%>" />
<img src="<%=(layoutURL)%>" />
```

getMetadataDocument

Used in: metadata
Parent elements: None

```
<aims:getMetadataDocument
    Attributes that set values:
    docId ="string"
    service ="string"

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    id ="object"
>

    No Child Elements
</aims:getMetadataDocument >
```

Bold: Attribute or child element is required.

Description:

Gets a metadata document for the given document ID.

Restrictions:

None

Notes:

None

Attribute descriptions for getMetadataDocument:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
docId	String used to uniquely identify a document. The client used to publish the metadata is responsible for creating the document ID. This ID is automatically assigned when using ArcCatalog. If another client is used, aims:getUUID can be used to request a valid ID. The format for an ID is the following: {HHHHHHHH-HHHH-HHHH-HHHH-HHHHHHHHHHHH} where H is a hexadecimal digit (0–9,a–f,A–F). The ID is limited to 38 characters.
id	Returns the Document object.
service	The name of the metadata service.

Examples for getMetadataDocument:

Example 1: Gets a handle to a metadata document for the given document ID.

```
<aims:getSettings id="hostName" value="meta_host_name" />
<aims:getSettings id="portNumber" value="meta_port_number" />
<aims:getSettings id="serviceName" value="meta_service_name" />

<aims:tcpConnection id="myConnection" host="<%= hostName %>" port="<%= portNumber %>"
debug="true" />

<% String docId = (String)request.getAttribute("docId");  %>

<aims:getMetadataDocument id="theDataset" connectionId="<%= myConnection %>" service="<%=
serviceName %>" docId="<%= docId %>" />

    <aims:datasetProperty id="theThumbnail" value="THUMBNAIL" dataset="<%= theDataset %>"
/>
    <aims:datasetProperty id="theGND" value="GND" dataset="<%= theDataset %>" />
    <aims:datasetProperty id="theServiceType" value="SERVICETYPE" dataset="<%= theDataset
%>" />
```

```
    <aims:datasetProperty id="hasSiblings" value="SIBLINGS" dataset="<%= theDataset %>"
/>
    <aims:datasetProperty id="theContentType" value="CONTENT" dataset="<%= theDataset %>"
/>
    <aims:datasetProperty id="theMapServer" value="SERVER" dataset="<%= theDataset %>" />
    <aims:datasetProperty id="theMapService" value="SERVICE" dataset="<%= theDataset %>"
/>

<P>
<img src="<%= theThumbnail %>" width="200" height="133" vspace="5" hspace="2" border="1">
<P>
Content Type: <%= theContentType %>
```

getRasterInfo

Used in: mapping
Parent elements: aims:map

```
<aims:getRasterInfo
    Attributes that set values:
    layerId ="string"
    x ="double"
    y ="double"

    Attributes that return values:
    id ="object"
>

    <coordSys... />

</aims:getRasterInfo >
```

Bold: Attribute or child element is required.

Description:

Returns the pixel value of an image at a given x,y coordinate location for the specified layer.

Restrictions:

None

Notes:

None

Attribute descriptions for getRasterInfo:

Attribute	Usage
id	Returns the RasterInfo object.
layerId	
x	X-coordinate of selected point.
y	Y-coordinate of selected point.

Examples for getRasterInfo:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myConnection%>"
name="imageServiceName"/>

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">

  <aims:getRasterInfo id="myRaster" layerId="0" x="1099.067" y="1349.42" />

  <aims:iterateRasterInfo rasterInfo="<%=(myRaster)%>" rasterId="id" number="no"
value="val">
    <b>Raster Id = </b> <%=(id)%>
    <b>Number    = </b> <%=(no)%>
    <b>Value     = </b> <%=(val)%><br>
  </aims:iterateRasterInfo>
</aims:map>
```

getResultAttribute

Used in: mapping
Parent elements: aims:iterateAddressMatchResults

```
<aims:getResultAttribute
    Attributes that set values:
    attribute ="VALUE | SCORE | POINT"

    Attributes that return values:
    id ="string"
>

    No Child Elements
</aims:getResultAttribute >
```

Bold: Attribute or child element is required.

Description:

Returns geocoding information.

Restrictions:

None

Notes:

None

Attribute descriptions for getResultAttribute:

Attribute	Usage
attribute	The name of the geocode property to retrieve.
id	Returns the value of the property specified in *id*.

Examples for getResultAttribute:

Example 1:

```
<aims:tcpConnection id="mytcpConnection" port="5300" host="myMachine" debug="true"/>
<aims:mapService id="myService" connectionId="<%=mytcpConnection%>" name="geoServiceName"
loadExtensions="true" />
<aims:map id="mapImage" serviceId="<%=myService%>" reset="false" >
  <aims:layer layerId="3" >
    <aims:addressMatchInputs id="myInputs" minScore="0" maxCandidates="100" >
      <aims:setInput inputId="STREET" value="380 New York St" />
      <aims:setInput inputId="ZONE" value="" />
      <aims:setInput inputId="CROSSSTREET" value="" />
    </aims:addressMatchInputs>
    <aims:iterateAddressMatchInputs inputs="<%=myInputs%>" count="myCount"
minScore="myMinScore" maxCandidates="myMax" style="myStyle" >
      <aims:getInputAttribute id="myDes" attribute="DESCRIPTION" />
      <aims:getInputAttribute id="myId" attribute="ID" />
      <aims:getInputAttribute id="myLabel" attribute="LABEL" />
      <aims:getInputAttribute id="myType" attribute="TYPE" />
      <aims:getInputAttribute id="myWidth" attribute="WIDTH" />
        <b>Description </b>: <%=myDes%> <br>
        <b>Id          </b>: <%=myId%> <br>
        <b>Label       </b>: <%=myLabel%> <br>
        <b>Type        </b>: <%=myType%> <br>
        <b>Width       </b>: <%=myWidth%> <br>
    </aims:iterateAddressMatchInputs>
    <aims:iterateAddressMatchResults inputs="<%=myInputs%>" count="resultsCount" >
      <aims:getResultAttribute id="myValue" attribute="VALUE" />
      <aims:getResultAttribute id="myScore" attribute="SCORE" />
      <aims:getResultAttribute id="myPoint" attribute="POINT" />
        <b>Value </b>: <%=myValue%> <br>
        <b>Score </b>: <%=myScore%> <br>
        <b>Point </b>: <%=myPoint%> <br>
    </aims:iterateAddressMatchResults>
    <b>Result count </b>: <%=resultsCount%>
  </aims:layer>
</aims:map>
```

getServer

Used in: administration
Parent elements: None

```
<aims:getServer
    Attributes that set values:
    name ="string"

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    id ="object"
    error ="string"
>

    No Child Elements
</aims:getServer >
```

Bold: Attribute or child element is required.

Description:

Returns a Server object that contains information on the containers and Virtual Server types associated with this server.

Restrictions:

None

Notes:

The Server object returned can be used in aims: iterateContainersForServer.

Attribute descriptions for getServer:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
error	Returns an error string if any occurred. Returns "Success" if no error.
id	Returns a Server object corresponding to the ArcIMS Spatial Server name.
name	The name of the server.

Examples for getServer:

Example 1:

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://brugge"
      role="admin"
      debug="true"
      username="myUsername"
      password="myPassword"
/>

<aims:getServer id="myServer" connectionId="<%=myhttpConnection%>" name="brugge"
error="myError1" />

<aims:iterateContainersForServer server="<%=myServer%>" count="containerCount" >
      <aims:getContainerAttribute id="myContainer" attribute="NAME"
connectionId="<%=myhttpConnection%>" virtualServers="myVirtualServers" />
      <td>
        <br><b>Container Name is: <%=myContainer%></b>
      </td>
      <aims:iterateVirtualServers virtualServers="<%=myVirtualServers%>"
count="serviceCount2" >
            <aims:getVirtualServerAttribute id="name" attribute="NAME" />
```

```
            <aims:getVirtualServerAttribute id="type" attribute="TYPE" />
            <aims:getVirtualServerAttribute id="access" attribute="ACCESS" />
            <aims:getVirtualServerAttribute id="description" attribute="DESCRIPTION" />
            <aims:getVirtualServerAttribute id="version" attribute="VERSION" />
            <aims:getVirtualServerAttribute id="containerset" attribute="CONTAINERSET" />
            <td>
              <br><b>name is: <%=name%></b>
              type is: <%=type%>
              description is: <%=description%>
              version is: <%=version%>
              access is: <%=access%>
              containerset is: <%=containerset%>
            </td>
      </aims:iterateVirtualServers>

</aims:iterateContainersForServer>
containerCount <%=containerCount%>
```

getService

Used in: administration
Parent elements: None

```
<aims:getService
    Attributes that set values:
    name ="string"

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    id ="object"
    error ="string"
>

    No Child Elements
</aims:getService >
```

Bold: Attribute or child element is required.

Description:

Returns the Service object of the named service.

Restrictions:

None

Notes:

None

Attribute descriptions for getService:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
error	Returns an error string if any occurred. Returns "Success" if no error.
id	Returns the Service object in the value of the *id* attribute.
name	The name of the service.

Examples for getService:

Example 1:

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://brugge"
      role="admin"
      username="admin"
      password="admin"
/>

<aims:getService id="myService" connectionId="<%=myhttpConnection%>" name="mexico2"
error="getError"/>

<aims:iterateServices services="<%=myService%>" count="myCount" >
  <aims:getServiceAttribute id="theName" attribute="NAME" />
  <aims:getServiceAttribute id="theFile" attribute="CONFIGFILE" />
  <aims:getServiceAttribute id="theType" attribute="IMAGETYPE" />
  <aims:getServiceAttribute id="theOutDir" attribute="OUTPUTDIR" />
  <aims:getServiceAttribute id="theOutClean" attribute="OUTPUTCLEANUP" />
  <aims:getServiceAttribute id="theOutURL" attribute="OUTPUTURL" />
  <aims:getServiceAttribute id="thePixel" attribute="PIXELCOUNT" />
  <aims:getServiceAttribute id="theVSName" attribute="VIRTUALSERVERNAME" />
  <aims:getServiceAttribute id="theVSInfo" attribute="VIRTUALSERVERINFO" />
```

```
  <aims:getServiceAttribute id="theStatus" attribute="STATUS" />
  <aims:getServiceAttribute id="theDes" attribute="DESCRIPTION" />
  <aims:getServiceAttribute id="theAcc" attribute="ACCESS" />
  <td>
  <br><b>Iterate one service</b>
  <br><b>Service name: <%=theName%></b>
  <br><b>Config file: <%=theFile%></b>
  <br><b>Image Type: <%=theType%></b>
  <br><b>Pixelcount: <%=thePixel%></b>
  <br><b>Virtual Server Name: <%=theVSName%></b>
  <br><b>Virtual Server Info: <%=theVSInfo%></b>
  <br><b>Output Dir: <%=theOutDir%></b>
  <br><b>Output URL: <%=theOutURL%></b>
  <br><b>Output Cleanup: <%=theOutClean%></b>
  <br><b>Description: <%=theDes%></b>
  <br><b>Access: <%=theAcc%></b>
  <br><b>Status: <%=theStatus%></b>
  <br><b></b>
  </td>
</aims:iterateServices>
```

getServiceAttribute

Used in: administration
Parent elements: aims:iterateServices

<aims:getServiceAttribute

Attributes that set values:
attribute ="NAME | CONFIGFILE | IMAGETYPE | OUTPUTCLEANUP | OUTPDIR | OUTPUTURL | PIXELCOUNT | ACCESS | STATUS | VIRTUALSERVERNAME | VIRTUALSERVERINFO"

Attributes that return values:
id ="*string*"
>

No Child Elements
</aims:getServiceAttribute >

Bold: Attribute or child element is required.

Description:

Returns a value defining an attribute of a service.

Restrictions:

None

Notes:

None

Attribute descriptions for getServiceAttribute:

Attribute	Usage
attribute	The name of the attribute to retrieve from the Service object. If attribute is not recognized, an empty string is returned.
id	Returns a string defining the attribute specified.

Examples for getServiceAttribute:

Example 1:

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://mymachine"
      role="admin"
      username="admin"
      password="admin"
/>

<aims:getServices id="myServices" connectionId="<%=myhttpConnection%>" />

<aims:iterateServices services="<%=myServices%>" count="myCount" >
  <aims:getServiceAttribute id="theName" attribute="NAME" />
  <aims:getServiceAttribute id="theFile" attribute="CONFIGFILE" />
  <aims:getServiceAttribute id="theType" attribute="IMAGETYPE" />
  <aims:getServiceAttribute id="theOutDir" attribute="OUTPUTDIR" />
  <aims:getServiceAttribute id="theOutClean" attribute="OUTPUTCLEANUP" />
  <aims:getServiceAttribute id="theOutURL" attribute="OUTPUTURL" />
  <aims:getServiceAttribute id="thePixel" attribute="PIXELCOUNT" />
  <aims:getServiceAttribute id="theVSName" attribute="VIRTUALSERVERNAME" />
  <aims:getServiceAttribute id="theVSInfo" attribute="VIRTUALSERVERINFO" />
  <aims:getServiceAttribute id="theStatus" attribute="STATUS" />
  <aims:getServiceAttribute id="theDes" attribute="DESCRIPTION" />
  <aims:getServiceAttribute id="theAcc" attribute="ACCESS" />
  <td>
```

```
  <br><b>Iterate all services</b>
  <br><b>Service name: <%=theName%></b>
  <br><b>Config file: <%=theFile%></b>
  <br><b>Image Type: <%=theType%></b>
  <br><b>Pixelcount: <%=thePixel%></b>
  <br><b>Virtual Server Name: <%=theVSName%></b>
  <br><b>Virtual Server Info: <%=theVSInfo%></b>
  <br><b>Output Dir: <%=theOutDir%></b>
  <br><b>Output URL: <%=theOutURL%></b>
  <br><b>Output Cleanup: <%=theOutClean%></b>
  <br><b>Description: <%=theDes%></b>
  <br><b>Access: <%=theAcc%></b>
  <br><b>Status: <%=theStatus%></b>
  <br><b></b>
  </td>
</aims:iterateServices>
```

getServices

Used in: administration
Parent elements: None

```
<aims:getServices
    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    id ="object"
    error ="string"
>

    No Child Elements
</aims:getServices >
```

Bold: Attribute or child element is required.

Description:

Returns a ServiceCollection object that contains a list of all services.

Restrictions:

None

Notes:

Use aims:iterateServices to retrieve individual Service objects for use in aims:adminService or to output information about all services.

Attribute descriptions for getServices:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
error	Returns an error string if any occurred. Returns "Success" if no error.
id	Returns the ServiceCollection object.

Examples for getServices:

Example 1:

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://mymachine"
      role="admin"
      username="admin"
      password="admin"
/>

<aims:getServices id="myServices" connectionId="<%=myhttpConnection%>" />

<aims:iterateServices services="<%=myServices%>" count="myCount" >
  <aims:getServiceAttribute id="theName" attribute="NAME" />
  <aims:getServiceAttribute id="theFile" attribute="CONFIGFILE" />
  <aims:getServiceAttribute id="theType" attribute="IMAGETYPE" />
  <aims:getServiceAttribute id="theOutDir" attribute="OUTPUTDIR" />
  <aims:getServiceAttribute id="theOutClean" attribute="OUTPUTCLEANUP" />
  <aims:getServiceAttribute id="theOutURL" attribute="OUTPUTURL" />
  <aims:getServiceAttribute id="thePixel" attribute="PIXELCOUNT" />
  <aims:getServiceAttribute id="theVSName" attribute="VIRTUALSERVERNAME" />
  <aims:getServiceAttribute id="theVSInfo" attribute="VIRTUALSERVERINFO" />
  <aims:getServiceAttribute id="theStatus" attribute="STATUS" />
  <aims:getServiceAttribute id="theDes" attribute="DESCRIPTION" />
  <aims:getServiceAttribute id="theAcc" attribute="ACCESS" />
```

```
  <td>
  <br><b>Iterate all services</b>
  <br><b>Service name: <%=theName%></b>
  <br><b>Config file: <%=theFile%></b>
  <br><b>Image Type: <%=theType%></b>
  <br><b>Pixelcount: <%=thePixel%></b>
  <br><b>Virtual Server Name: <%=theVSName%></b>
  <br><b>Virtual Server Info: <%=theVSInfo%></b>
  <br><b>Output Dir: <%=theOutDir%></b>
  <br><b>Output URL: <%=theOutURL%></b>
  <br><b>Output Cleanup: <%=theOutClean%></b>
  <br><b>Description: <%=theDes%></b>
  <br><b>Access: <%=theAcc%></b>
  <br><b>Status: <%=theStatus%></b>
  <br><b></b>
  </td>
</aims:iterateServices>
```

getSettings

Used in: mapping metadata administration
Parent elements: None

```
<aims:getSettings
    Attributes that set values:
    value ="string"
    propFileName ="string"

    Attributes that return values:
    id ="string"
>

    No Child Elements
</aims:getSettings >
```

Bold: Attribute or child element is required.

Description:

Reads setting values from a property file.

Restrictions:

None

Notes:

None

Attribute descriptions for getSettings:

Attribute	Usage
id	A variable name that stores a setting.
propFileName	Name of the property file.
value	The name of the value to read from the properties file.

Examples for getSettings:

Example 1:

```
<aims:getSettings id="hostName" value="meta_host_name" />

or

<aims:getSettings id="hostName" value="meta_host_name" propFileName="propFile.properties"
/>
```

getStartupParameter

Used in: administration
Parent elements: None

```
<aims:getStartupParameter
    Attributes that set values:
    parameter ="recyclableservers | singlethreadedservers | defaultrecyclingfrequency |
defaultrecyclingreferencehour | defaultrecyclingreferenceminute"

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    id ="object"
    error ="string"
>

    No Child Elements
</aims:getStartupParameter >
```

Bold: Attribute or child element is required.

Description:

Retrieves values from the ArcIMS Application Server for default Virtual Server recycling and the status of which Virtual Servers are single threaded.

Restrictions:

None

Notes:

None

Attribute descriptions for getStartupParameter:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
error	Returns an error string if any occurred. Returns "Success" if no error.
id	Returns a string value of the *parameter* requested.
parameter	The startup parameter for a Virtual Server. • recyclableservers: A comma separated list of Virtual Server types to be recycled. By default, only ArcMap Servers are set as recyclable. • singlethreadedservers: A comma separated list of Virtual Server types that are single threaded. By default, only ArcMap Servers are set as single threaded. • defaultrecyclingfrequency: number of times in a day to recycle. Valid values are 0, 1, 2, 3, 4, 6, 12, and 24. A value of 0 means no recycling. • defaultrecyclingreferencehour: the hour to start recycling (0–23, where 0 is midnight). • defaultrecyclingreferenceminute: the minute to start recycling. Valid values are 0, 15, 30, and 45.

Examples for getStartupParameter:

Example 1:

```
<aims:httpConnection id="myhttpConnection" url="http://mymachine" role="admin"
debug="true" username="admin" password="admin" />

<aims:getStartupParameter id="recycleServers" connectionId="<%=myhttpConnection%>"
parameter="recyclableservers" error="error4" />
recycleServers <%=recycleServers%>
get parameter error <%=error4%>

<aims:getStartupParameter id="frequency" connectionId="<%=myhttpConnection%>"
parameter="defaultrecyclingfrequency" />
recycleFrequency <%=frequency%>
```

getUUID

Used in: metadata
Parent elements: None

```
<aims:getUUID
    Attributes that set values:
    service ="string"
    count ="integer"  [1]

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    id ="object"
>

    No Child Elements
</aims:getUUID >
```

Bold: Attribute or child element is required.

Description:

Generates a unique ID that can be used as a document ID.

Restrictions:

None

Notes:

None

Attribute descriptions for getUUID:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
count	Determines the number of unique IDs to generate.
id	Returns a vector of UUID strings. The number of UUID strings returned is determined by the count attribute. The format for a UUID is the following: {HHHHHHHH-HHHH-HHHH-HHHH-HHHHHHHHHHHH} where H is a hexadecimal digit (0–9,a–f,A–F). The ID is limited to 38 characters.
service	The name of the metadata service to generate the UUID strings.

Examples for getUUID:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true"
username="publish" password="publish" />

<aims:getUUID id="myUUID" connectionId="<%=(myConnection)%>" service="serviceName"
count="3" />
UUID : = <%=(myUUID.elementAt(0))%>
```

getVirtualServer

Used in: administration
Parent elements: None

```
<aims:getVirtualServer
    Attributes that set values:
    name ="string"

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    id ="object"
    error ="string"
>

    No Child Elements
</aims:getVirtualServer >
```

Bold: Attribute or child element is required.

Description:

Returns the VirtualServer object of the named Virtual Server.

Restrictions:

None

Notes:

None

Attribute descriptions for getVirtualServer:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
error	Returns an error string if any occurred. Returns "Success" if no error.
id	Returns the VirtualServer object.
name	The name of the Virtual Server.

Examples for getVirtualServer:

Example 1:

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://mymachine"
      role="admin"
      debug="true"
      username="admin"
      password="admin"
/>

<aims:getVirtualServer id="myServer" connectionId="<%=myhttpConnection%>"
name="FeatureServer1" error="getError" />

<aims:iterateVirtualServers virtualServers="<%=myServer%>" count="serverCount" >
      <aims:getVirtualServerAttribute id="name" attribute="NAME" />
      <aims:getVirtualServerAttribute id="type" attribute="TYPE" />
      <aims:getVirtualServerAttribute id="description" attribute="DESCRIPTION" />
      <aims:getVirtualServerAttribute id="version" attribute="VERSION" />
      <aims:getVirtualServerAttribute id="access" attribute="ACCESS" />
      <aims:getVirtualServerAttribute id="containerset" attribute="CONTAINERSET" />
      <td>
        <br><b>name is: <%=name%></b>
```

```
        type is: <%=type%>
        description is: <%=description%>
        version is: <%=version%>
        access is: <%=access%>
        containerset is: <%=containerset%>
      </td>
</aims:iterateVirtualServers>
```

getVirtualServerAttribute

Used in: administration
Parent elements: aims:iterateVirtualServers

<aims:getVirtualServerAttribute

Attributes that set values:
attribute ="NAME | ACCESS | DESCRIPTION | TYPE | VERSION | CONTAINERSET | RECYCLE_FREQUENCY | RECYCLE_HOUR | RECYCLE_MINUTE"

Attributes that return values:
id ="*string*"
>

No Child Elements
</aims:getVirtualServerAttribute >

Bold: Attribute or child element is required.

Description:

Returns a string defining an attribute of a Virtual Server.

Restrictions:

None

Notes:

None

Attribute descriptions for getVirtualServerAttribute:

Attribute	Usage
attribute	The name of the attribute to retrieve from the VirtualServer object. If attribute value is not recognized, an empty string is returned. • NAME: Specifies the Virtual Server name. • ACCESS: Specifies whether a Virtual Server is public or private. In general, "PUBLIC" is used for Image, Feature, ArcMap, and Metadata Servers. "PRIVATE" is used for Query, Geocode, and Extract Servers. • CONTAINERSET: Comma-delimited list of Spatial Servers or containers associated with the Virtual Server followed by a space and the number of instances currently running (that is, mymachine_1 2, mymachine_2 1). • DESCRIPTION: Provides a description of the Virtual Server. • RECYCLE_FREQUENCY: Number of times per each 24-hour period to recycle all Spatial Servers associated with this Virtual Server. If this value is 0, the Virtual Server does not implement recycling. • RECYCLE_HOUR: Starting hour to begin the first recycling period each day using a clock. "0" is midnight. • RECYCLE_MINUTE: Starting minute to begin the first recycling period each day using a 24-hour clock. • TYPE: The Virtual Server type: ImageServer, FeatureServer, MetadataServer, ExtractServer, GeocodeServer, QueryServer. For ArcMap Server, "ImageServer" is returned for this attribute. • VERSION: Specifies whether an ImageServer is an ArcMap Image Server. For ArcMap Image Servers, "ArcMap" is returned. For all other server types, "" is returned.
id	Returns a string defining the attribute specified.

Examples for getVirtualServerAttribute:

Example 1:

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://mymachine"
      role="admin"
      debug="true"
      username="admin"
      password="admin"
/>

<aims:getVirtualServers id="myServers" connectionId="<%=myhttpConnection%>"
error="getError" />

<aims:iterateVirtualServers virtualServers="<%=myServers%>" count="serverCount" >
      <aims:getVirtualServerAttribute id="name" attribute="NAME" />
      <aims:getVirtualServerAttribute id="type" attribute="TYPE" />
      <aims:getVirtualServerAttribute id="access" attribute="ACCESS" />
      <aims:getVirtualServerAttribute id="description" attribute="DESCRIPTION" />
      <aims:getVirtualServerAttribute id="version" attribute="VERSION" />
      <aims:getVirtualServerAttribute id="containerset" attribute="CONTAINERSET" />
      <aims:getVirtualServerAttribute id="frequency" attribute="RECYCLE_FREQUENCY" />
      <aims:getVirtualServerAttribute id="hour" attribute="RECYCLE_HOUR" />
      <aims:getVirtualServerAttribute id="minute" attribute="RECYCLE_MINUTE" />
      <td>
        <br><b>name is: <%=name%></b><BR>
        type is: <%=type%><BR>
        description is: <%=description%><BR>
        version is: <%=version%><BR>
        access is: <%=access%><BR>
        containerset is: <%=containerset%><BR>
        frequency is: <%=frequency%><BR>
        hour is: <%=hour%><BR>
        minute is: <%=minute%>
      </td>
</aims:iterateVirtualServers>
```

getVirtualServers

Used in: administration
Parent elements: None

```
<aims:getVirtualServers
    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    id ="object"
    error ="string"
>

    No Child Elements
</aims:getVirtualServers >
```

Bold: Attribute or child element is required.

Description:

Returns a VirtualServerCollection object that contains all Virtual Servers.

Restrictions:

None

Notes:

Use aims:iterateVirtualServers to output information about all Virtual Servers.

Attribute descriptions for getVirtualServers:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
error	Returns an error string if any occurred. Returns "Success" if no error.
id	Returns the VirtualServerCollection object.

Examples for getVirtualServers:

Example 1:

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://mymachine"
      role="admin"
      debug="true"
      username="admin"
      password="admin"
/>

<aims:getVirtualServers id="myServers" connectionId="<%=myhttpConnection%>"
error="getError" />

<aims:iterateVirtualServers virtualServers="<%=myServers%>" count="serverCount" >
      <aims:getVirtualServerAttribute id="name" attribute="NAME" />
      <aims:getVirtualServerAttribute id="type" attribute="TYPE" />
      <aims:getVirtualServerAttribute id="access" attribute="ACCESS" />
      <aims:getVirtualServerAttribute id="description" attribute="DESCRIPTION" />
      <aims:getVirtualServerAttribute id="version" attribute="VERSION" />
      <aims:getVirtualServerAttribute id="containerset" attribute="CONTAINERSET" />
      <td>
        <br><b>name is: <%=name%></b>
        type is: <%=type%>
        description is: <%=description%>
        version is: <%=version%>
        access is: <%=access%>
        containerset is: <%=containerset%>
      </td>
</aims:iterateVirtualServers>
```

gradientFillSymbol

Used in: mapping
Parent elements: aims:buffer aims:displayFeatures aims:exact aims:other aims:polygon aims:range aims:simpleRenderer aims:targetLayer

<aims:gradientFillSymbol

Attributes that set values:

When using ArcMap Server:
finishColor ="0,0,0 - 255,255,255" **[0,255,0]**
startColor ="0,0,0 - 255,255,255" **[255,0,0]**
type ="bdiagonal | fdiagonal | horizontal | vertical" **[bdiagonal]**

When using Image Server:
antialiasing ="true | false" **[false]**
finishColor ="0,0,0 - 255,255,255" **[0,255,0]**
overlap ="true | false" **[true]**
startColor ="0,0,0 - 255,255,255" **[255,0,0]**
transparency ="0.0 - 1.0" **[1.0]**
type ="bdiagonal | fdiagonal | horizontal | vertical" **[bdiagonal]**
>

No Child Elements
</aims:gradientFillSymbol >

Description:

Fills the polygon with a gradual gradient based on two colors.

Restrictions:

In ArcMap Image Services, symbol is valid only in acetate layers.

Notes:

None

Attribute descriptions for gradientFillSymbol:

Attribute	Usage
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
finishColor	End color using RGB values.
overlap	Determines if labels can overlap this symbol. When "true", labels can overlap. When "false", labels will not overlap the symbol. If labels are not drawing as expected, check if *overlap* is set to "false" for this symbol or any other symbol in the service.
startColor	Start color using RGB values.
transparency	Value to set percentage of transparency. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.
type	Symbol type.

Examples for gradientFillSymbol:

Example 1:

```
<aims:tcpConnection id="mytcpConnection" port="5300" host="myMachine" />
<aims:mapService id="myMapService" connectionId="<%=mytcpConnection%>"
name="serviceName"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:createLayer layerId="idPolygon" name="polygon" type="acetate" visible="true">
    <aims:acetateObject units="pixel">
      <aims:polygon coords="10 110;400 110;400 160;10 160;10 110">
        <aims:gradientFillSymbol transparency="1.0" type="vertical" startColor="0,255,0"
finishColor="0,0,255" overlap="true" />
      </aims:polygon>
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>
<img src="<%=(myMap)%>" />
```

groupRenderer

Used in: mapping
Parent elements: aims:createLayer aims:groupRenderer aims:layer aims:scaleDependentRenderer

<groupRenderer >
No Attributes

(m) <groupRenderer... />
(m) <scaleDependentRenderer... />
(m) <simpleLabelRenderer... />
(m) <simpleRenderer... />
(m) <valueMapLabelRenderer... />
(m) <valueMapRenderer... />

</aims:groupRenderer >

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Groups two or more renderers together. Common uses are to group feature rendering and labeling, to use multiple scale-dependent renderers on the same layer, and to create complex symbology.

Restrictions:

- At least one nested tag is required. Any combination and number of listed nested tags can be used.
- Not valid with ArcMap Server.

Notes:

None

Examples for groupRenderer:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />
<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:layer layerId="4" visible="true" >
    <aims:groupRenderer>
      <aims:simpleRenderer>
        <aims:simpleLineSymbol color="150,0,255" />
      </aims:simpleRenderer>
      <aims:valueMapLabelRenderer lookUpField="NAME" labelField="FCC">
        <aims:exact label="Colorado Crk" value="COLORADO CREEK">
          <aims:textSymbol/>
        </aims:exact>
        <aims:exact label="Isabel Crk" value="ISABEL CREEK">
          <aims:textSymbol fontSize="30" font="Times New Roman" fontColor="0,255,0"/>
        </aims:exact>
     </aims:valueMapLabelRenderer>
    </aims:groupRenderer>
  </aims:layer>
</aims:map>
<img src="<%=(myMap)%>" />
```

hashLineSymbol

Used in: mapping
Parent elements: aims:buffer aims:displayFeatures aims:exact aims:line aims:other aims:polygon aims:range aims:simpleRenderer aims:targetLayer

<aims:hashLineSymbol

Attributes that set values:

When using ArcMap Server:
color ="0,0,0 - 255,255,255" **[0,0,0]**
interval ="0 - NNN" **[8]**
lineThickness ="1 - NNN" **[1]**
tickThickness ="1 - NNN" **[1]**
type ="foreground | background" **[foreground]**
width ="1 - NNN" **[6]**

When using Image Server:
antialiasing ="true | false" **[false]**
color ="0,0,0 - 255,255,255" **[0,0,0]**
interval ="0 - NNN" **[8]**
lineThickness ="1 - NNN" **[1]**
overlap ="true | false" **[true]**
tickThickness ="1 - NNN" **[1]**
transparency ="0.0 - 1.0" **[1.0]**
type ="foreground | background" **[foreground]**
width ="1 - NNN" **[6]**
>

No Child Elements
</aims:hashLineSymbol >

Description:

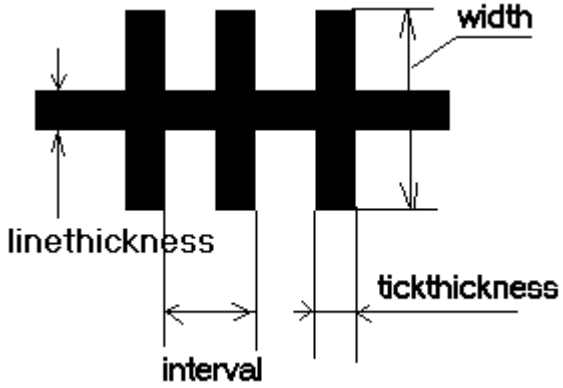
Line symbol for drawing railroad symbols.

Restrictions:

In ArcMap Image Services, symbol is valid only in acetate layers.

Notes:

- The following figure shows the different components that make up aims:hashLineSymbol:



- A smoothing algorithm is used on the line to get a better hash effect. If a different line symbol is used, the smoothing algorithm is not applied, and a line will not overlay exactly with a hashline (railroad). In order to make a line overlay the hashline, use aims:hashLineSymbol for the line and set the attribute *type* to "background".

Attribute descriptions for hashLineSymbol:

Attribute	Usage
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
color	Symbol color using RGB values.
interval	Distance between railroad crosshatches in pixels.

Attribute	Usage
lineThickness	Line thickness in pixels.
overlap	Determines if labels can overlap this symbol. When "true", labels can overlap. When "false", labels will not overlap the symbol. If labels are not drawing as expected, check if *overlap* is set to "false" for this symbol or any other symbol in the service.
tickThickness	Tick thickness in pixels.
transparency	Value to set percentage of transparency. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.
type	If *type* is "foreground", the symbol draws as a railroad with the crosshash. If *type* is "background", the symbol draws as a simple line without the crosshash.
width	Width of the crosshash segments in pixels.

Examples for hashLineSymbol:

Example 1:

```
<aims:tcpConnection id="mytcpConnection" port="5300" host="myMachine"/>
<aims:mapService id="myMapService" connectionId="<%=mytcpConnection%>"
name="serviceName"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:separators  ts=";" cs=" " />
  <aims:createLayer layerId="myLayerId" name="line" type="acetate" visible="true">
    <aims:acetateObject units="pixel">
      <aims:line coords="10 150;400 150" >
        <aims:hashLineSymbol color="0,0,0" lineThickness="8" tickThickness="8"
transparency="0.5" interval="16" width="16" type="foreground" antialiasing="false"
overlap="true" />
      </aims:line>
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>
<img src="<%=(myMap)%>" />
```

hideLayer

Used in: mapping
Parent elements: aims:legend

```
<aims:hideLayer
    Attributes that set values:
    hide ="true | false"
    layerId ="string"
>
    No Child Elements
</aims:hideLayer >
```

Bold: Attribute or child element is required.

Description:

Hides a specified layer from a legend when the legend is displayed.

Restrictions:

None

Notes:

None

Attribute descriptions for hideLayer:

Attribute	Usage
hide	Toggle to hide layer. A value of "true" hides the layer from the legend.
layerId	Layer ID defined in map configuration file or from layer added using aims:createLayer.

Examples for hideLayer:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />
<aims:mapService id="myService" connectionId="<%=myTCPConnection%>" name="serviceName"/>
<aims:map id="myMapURL" serviceId="<%=myService%>" legend="myLegend" >
  <aims:legend title="Legend" font="Arial" autoExtend="true" columns="1" width="170"
height="100" backgroundColor="255,255,0" >
    <aims:hideLayer layerId="6" hide="true" />
  </aims:legend>
</aims:map>
<img src="<%=myLegend%>" />
```

hole

Used in: mapping
Parent elements: aims:polygon

```
<aims:hole
    Attributes that set values:
    coords ="x1 y1;...xn yn"
>

    No Child Elements
</aims:hole >
```

Bold: Attribute or child element is required.

Description:

Provides the x,y coordinate locations for holes inside a polygon feature.

Restrictions:

The polygon that makes up a hole must be closed.

Notes:

None

Attribute descriptions for hole:

Attribute	Usage
coords	X,y coordinates representing one or more holes inside a polygon. Coordinate x,y values are separated by white space, and coordinate pairs are separated by a semicolon by default. The separators can be changed by using aims:separators. Each hole must be closed.

Examples for hole:

Example 1:

```
<aims:tcpConnection id="myConnection" host="Kabilan" port="5300" debug="true" />
<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="SantaClara"/>
<aims:map id="mapImage" serviceId="<%=myMapService%>" width="500" height="300" >

  <aims:separators ts=";" cs="," />

  <aims:createLayer layerId="polygonLayer" type="acetate" visible="true" >
    <aims:acetateObject>
      <aims:polygon coords="-122.038,37.366;-121.575,37.366;-121.575,37.218;-
122.038,37.218;-122.038,37.366">
            <aims:gradientFillSymbol />
       <aims:hole coords="-121.897,37.338;-121.812,37.338;-121.812,37.307;-
121.897,37.307;-121.897,37.338" />
      </aims:polygon>
    </aims:acetateObject>
       <aims:acetateObject units="pixel" >
      <aims:polygon coords="10,10;400,10;400,100;10,100;10,10">
        <aims:simplePolygonSymbol fillColor="0,255,0" transparency="1.0" />
        <aims:hole coords="150,40;150,60;200,60;200,40;150,40" />
      </aims:polygon>
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>
<img src="<%=(mapImage)%>" />
```

httpConnection

Used in: **mapping** **metadata** **administration**
Parent elements: None

```
<aims:httpConnection

    Attributes that set values:
    url ="string"
    debug ="true | false"  [false]
    password ="string"
    role ="user | admin"  [user]
    username ="string"

    Attributes that return values:
    id ="object"
    ping ="OK | FAIL | FIRST_LOGIN"
>

    No Child Elements
</aims:httpConnection >
```

Bold: Attribute or child element is required.

Description:

Creates an HTTP or HTTPS connection proxy to use with aims:map.

Restrictions:

When *role* is "admin", the attribute *debug* can only be set to "false".

Notes:

None

Attribute descriptions for httpConnection:

Attribute	Usage
debug	Shows debug messages in servlet's log file.
id	Returns ConnectionProxy object.
password	Password when service authentication is used.
ping	Returns a string showing the status from pinging the ArcIMS Application Server.
role	Defines the role of the connection. For administration requests, *role* must be set to "admin".
url	The URL of the machine (i.e., http://mymachine.domain.com).
username	Username when service authentication is used.

Examples for httpConnection:

Example 1:

```
<aims:httpConnection id="myhttpConnection" url="http://myMachine.domain.com" debug="true"
/>

<aims:mapService id="myService" connectionId="<%=myhttpConnection%>" name="serviceName"
/>

<aims:map id="mapImage" serviceId="<%=myService%>" width="500" height="400" />
<img src="<%=(mapImage)%>" />
```

imageWorkspace

Used in: mapping
Parent elements: aims:createLayer

```
<aims:imageWorkspace
    Attributes that set values:
    directory ="string"
    name ="string"
>

    No Child Elements
</aims:imageWorkspace >
```

Bold: Attribute or child element is required.

Description:

Defines the workspace of an image data source.

Restrictions:

- Must refer to an existing data source.
- Not valid with ArcMap Server.

Notes:

- It is recommended not to use aims:imageWorkspace in a request. All workspaces should be included in the map configuration file.
- To add an image workspace, <MAP dynamic="true" > must be set in the map configuration file.

Attribute descriptions for imageWorkspace:

Attribute	Usage
directory	Directory containing images. UNC pathnames can be used (\\myComputer\shapefiledirectory).
name	Workspace name. Must be unique among all data sources.

Examples for imageWorkspace:

Example 1:

```
<aims:tcpConnection id="mytcpConnection" port="5300" host="myMachine" debug="true" />
<aims:mapService id="myService" connectionId="<%=mytcpConnection%>" name="serviceName" />

<aims:map id="mapImage" serviceId="<%=myService%>" width="500" height="300" >
  <aims:createLayer layerId="imageLayer" type="image" name="SanFranciscoImage"
visible="true" >
    <aims:imageWorkspace name="workspace2"
directory="C:\\ArcIMS\\AXL\\TutorialData\\SanFrancisco" />
    <aims:dataset name="sanfran.tif" type="image" workspace="workspace2" />
  </aims:createLayer>
</aims:map>
<img src="<%=(mapImage)%>" />
```

iterateAddressMatchInputs

Used in: mapping
Parent elements: aims:layer

<aims:iterateAddressMatchInputs

Attributes that pass objects:
inputs ="*object*"

Attributes that return values:
count ="*integer*"
maxCandidates ="*integer*" **[20]**
minScore ="0 – 100" **[60]**
style ="USAddressZ | USAddress | USSingleHouse | USSingleHouseZ | USSingleRange | USSingleRangeZ | Zip4 | Zip4Range | Zip5 | SingleField"
>

(m) <getInputAttribute... />

</aims:iterateAddressMatchInputs >

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Iterates through items in the addressMatchInput object.

Restrictions:

None

Notes:

ArcIMS has 10 standard address styles:

GCSTYLE Address Style	Description
USAddressZ	U.S. streets with zone: This style considers street addresses with zone information, such as postal ZIP Codes or city names. Street features are represented by two house number intervals, one for the left side of the street and the other for the right side of the street. The parity (odd/even) of each interval should agree. The beginning number of the intervals can either be lower or higher than the end number.
USAddress	U.S. streets: This style is used for address geocoding without zone information. Street features are represented by two house number intervals, one for the left side of the street and the other for the right side of the street.
USSingleRangeZ	U.S. single range with zone: This style considers street or parcel addresses with zone information, such as postal ZIP Codes or city names. Features are represented by one house number interval. No parity (odd/even) of the house numbers is required.
USSingleRange	U.S. single range: This style is used for address geocoding without zone information. Features are represented by one house number interval. No parity (odd/even) of the house numbers is required.
USSingleHouseZ	U.S. single house with zone: This style considers parcel or point addresses with zone information, such as postal ZIP Codes or city names. Features are represented by a single house number.
USSingleHouse	U.S. single house: This style is used for address geocoding without zone information. Features are represented by a single house number.
Zip4	ZIP+4: This style is used for geocoding ZIP+4 using two components (ZIP and ZIP4).
Zip4Range	ZIP+4 range: This style is used for geocoding ZIP+4 in a range using three components (ZIP, ZIP4Low, and ZIP4High).
Zip5	ZIP 5-digit: This style is used for geocoding five-digit ZIP Code addresses.
SingleField	Single field: This style is used for geocoding based on the value of a single field in the reference layer's attribute table. The style contains one component (Keyfield).

Attribute descriptions for iterateAddressMatchInputs:

Attribute	Usage
count	Number of objects in the Results collection.
inputs	AddressMatchInputs Object retrieved from aims:addressMatchInputs *id* attribute. Results are retrieved from the AddressMatchInputs object.
maxCandidates	Maximum number of returned candidates.
minScore	Minimum score of returned candidates. If not included, all candidates with scores above 60 are returned. A candidate with a score of 100 means a perfect match, and 0 means no match.
style	Type of address style.

Examples for iterateAddressMatchInputs:

Example 1:

```
<aims:tcpConnection id="mytcpConnection" port="5300" host="myMachine" debug="true"/>
<aims:mapService id="myService" connectionId="<%=mytcpConnection%>" name="geoServiceName"
loadExtensions="true" />
<aims:map id="mapImage" serviceId="<%=myService%>" reset="false" >
  <aims:layer layerId="3" >
    <aims:addressMatchInputs id="myInputs" minScore="0" maxCandidates="100" >
      <aims:setInput inputId="STREET" value="380 New York St" />
      <aims:setInput inputId="ZONE" value="" />
      <aims:setInput inputId="CROSSSTREET" value="" />
    </aims:addressMatchInputs>
    <aims:iterateAddressMatchInputs inputs="<%=myInputs%>" count="myCount"
minScore="myMinScore" maxCandidates="myMax" style="myStyle" >
      <aims:getInputAttribute id="myDes" attribute="DESCRIPTION" />
      <aims:getInputAttribute id="myId" attribute="ID" />
      <aims:getInputAttribute id="myLabel" attribute="LABEL" />
      <aims:getInputAttribute id="myType" attribute="TYPE" />
      <aims:getInputAttribute id="myWidth" attribute="WIDTH" />
        <b>Description </b>: <%=myDes%> <br>
        <b>Id          </b>: <%=myId%> <br>
```

```
        <b>Label       </b>: <%=myLabel%> <br>
        <b>Type        </b>: <%=myType%> <br>
        <b>Width       </b>: <%=myWidth%> <br>
    </aims:iterateAddressMatchInputs>
    <aims:iterateAddressMatchResults inputs="<%=myInputs%>" count="resultsCount" >
      <aims:getResultAttribute id="myValue" attribute="VALUE" />
      <aims:getResultAttribute id="myScore" attribute="SCORE" />
      <aims:getResultAttribute id="myPoint" attribute="POINT" />
        <b>Value </b>: <%=myValue%> <br>
        <b>Score </b>: <%=myScore%> <br>
        <b>Point </b>: <%=myPoint%> <br>
    </aims:iterateAddressMatchResults>
    <b>Result count </b>: <%=resultsCount%>
  </aims:layer>
</aims:map>
```

iterateAddressMatchResults

Used in: mapping
Parent elements: aims:layer

```
<aims:iterateAddressMatchResults
    Attributes that pass objects:
    inputs ="object"

    Attributes that return values:
    count ="integer"
>
    (m) <getResultAttribute... />
</aims:iterateAddressMatchResults >
```

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Iterates through address match results after values are set in aims:addressMatchInputs.

Restrictions:

None

Notes:

Aims:mapService must have *loadExtensions="true"* to populate aims:addressMatchInputs.

Attribute descriptions for iterateAddressMatchResults:

Attribute	Usage
count	Number of objects in the Results collection.
inputs	AddressMatchInputs object retrieved from aims:addressMatchInputs *id* attribute. Results are retrieved from the AddressMatchInputs object.

Examples for iterateAddressMatchResults:

Example 1:

```
<aims:tcpConnection id="mytcpConnection" port="5300" host="myMachine" debug="true"/>
<aims:mapService id="myService" connectionId="<%=mytcpConnection%>" name="geoServiceName"
loadExtensions="true" />
<aims:map id="mapImage" serviceId="<%=myService%>" reset="false" >
  <aims:layer layerId="3" >
    <aims:addressMatchInputs id="myInputs" minScore="0" maxCandidates="100" >
      <aims:setInput inputId="STREET" value="380 New York St" />
      <aims:setInput inputId="ZONE" value="" />
      <aims:setInput inputId="CROSSSTREET" value="" />
    </aims:addressMatchInputs>
    <aims:iterateAddressMatchInputs inputs="<%=myInputs%>" count="myCount"
minScore="myMinScore" maxCandidates="myMax" style="myStyle" >
      <aims:getInputAttribute id="myDes" attribute="DESCRIPTION" />
      <aims:getInputAttribute id="myId" attribute="ID" />
      <aims:getInputAttribute id="myLabel" attribute="LABEL" />
      <aims:getInputAttribute id="myType" attribute="TYPE" />
      <aims:getInputAttribute id="myWidth" attribute="WIDTH" />
        <b>Description </b>: <%=myDes%> <br>
        <b>Id          </b>: <%=myId%> <br>
        <b>Label       </b>: <%=myLabel%> <br>
        <b>Type        </b>: <%=myType%> <br>
        <b>Width       </b>: <%=myWidth%> <br>
    </aims:iterateAddressMatchInputs>
    <aims:iterateAddressMatchResults inputs="<%=myInputs%>" count="resultsCount" >
      <aims:getResultAttribute id="myValue" attribute="VALUE" />
      <aims:getResultAttribute id="myScore" attribute="SCORE" />
      <aims:getResultAttribute id="myPoint" attribute="POINT" />
        <b>Value </b>: <%=myValue%> <br>
        <b>Score </b>: <%=myScore%> <br>
        <b>Point </b>: <%=myPoint%> <br>
    </aims:iterateAddressMatchResults>
    <b>Result count </b>: <%=resultsCount%>
  </aims:layer>
</aims:map>
```

iterateContainersForServer

Used in: administration
Parent elements: None

<aims:iterateContainersForServer

Attributes that pass objects:
server ="*object*"

Attributes that return values:
count ="*integer*"
>

(m) <getContainerAttribute... />

</aims:iterateContainersForServer >

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Iterates through all ArcIMS Spatial Servers on a server machine to display information about each server.

Restrictions:

None

Notes:

Use aims:getContainerAttribute to retrieve values about containers from a server.

Attribute descriptions for iterateContainersForServer:

Attribute	Usage
count	Returns the number of containers iterated.
server	A Server object retrieved from aims:getServer.

Examples for iterateContainersForServer:

Example 1:

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://brugge"
      role="admin"
      debug="true"
      username="admin"
      password="admin"
/>

aims:getServer id="myServer" connectionId="<%=myhttpConnection%>" name="brugge" error="myError1" />

<aims:iterateContainersForServer server="<%=myServer%>" count="containerCount" >
      <aims:getContainerAttribute id="myContainer" attribute="NAME"
connectionId="<%=myhttpConnection%" virtualServers="myVirtualServers" />
      <td>
        <br><b>Container Name is: <%=myContainer%></b>
      </td>
      <aims:iterateVirtualServers virtualServers="<%=myVirtualServers%>" count="serviceCount2" >
            <aims:getVirtualServerAttribute id="name" attribute="NAME" />
            <aims:getVirtualServerAttribute id="type" attribute="TYPE" />
            <aims:getVirtualServerAttribute id="access" attribute="ACCESS" />
            <aims:getVirtualServerAttribute id="description" attribute="DESCRIPTION" />
            <aims:getVirtualServerAttribute id="version" attribute="VERSION" />
            <aims:getVirtualServerAttribute id="containerset" attribute="CONTAINERSET" />
            <td>
              <br><b>name is: <%=name%></b>
              type is: <%=type%>
              description is: <%=description%>
              version is: <%=version%>
              access is: <%=access%>
              containerset is: <%=containerset%>
            </td>
      </aims:iterateVirtualServers>

</aims:iterateContainersForServer>
containerCount <%=containerCount%>
```

iterateContentType

Used in: metadata
Parent elements: None

```
<aims:iterateContentType

    Attributes that pass objects:
    value ="object"

    Attributes that return values:
    count ="integer"
    id ="string"
    result ="true | false"
>

    No Child Elements
</aims:iterateContentType >
```

Bold: Attribute or child element is required.

Description:

Iterates through metadata content types and their counts.

Restrictions:

None

Notes:

None

Attribute descriptions for iterateContentType:

Attribute	Usage
count	The count of the current content type in the iteration.
id	The name of the current content type in the iteration. Can be any string.
result	Determines whether there is more than one content type present in the result set.
value	The result set from which to generate the summary.

Examples for iterateContentType:

Example 1: Iterates through metadata content types and their counts.

```
<%-- Get a handle to the results --%>
<% DatasetCollection results = (DatasetCollection)request.getAttribute("results");


<%-- Start: Type of Content Summary --%>

<% boolean printedTableHeader = false; %>
<aims:iterateContentType id="currentContent" count="contentCount" value="<%=results%>"
result="moreThanOne">

<%-- check how many content types there are --%>
<% if (moreThanOne.equals("true")) { %>

      <% if (!printedTableHeader)      { // print out a table header, since we are on the
first result %>

            <% if (goTo.equals("browse")) { %>
                  <br>
            <% } %>

            <a name="top">
```

```
            <%-- print out table column headings --%>
            <table width="100%" border="0" cellspacing="0" cellpadding="3">
            <tr>
                  <td><span class="explorerText"><b>Type of Content</b></span></td>
                  <td align="center"><span
class="explorerText"><b>Records</b></span></td>
            </tr>

            <% printedTableHeader = true; %>

      <% } %>

      <%-- iterate through all the content types --%>
      <tr>
            <td><a href="#<%=currentContent%>"><%=currentContent%></a></td>
            <td align="center"><span class="explorerText"><%=contentCount%></span></td>
      </tr>

<% } %>

</aims:iterateContentType>

<% if (printedTableHeader) { // close the table %>
      </table>
      <br>
<% } %>

<%-- End: Type of Content Summary --%>
```

iterateDataframe

Used in: mapping
Parent elements: aims:getArcMapLayout

```
<aims:iterateDataframe

    Attributes that pass objects:
    dataframeCollection ="object"

    Attributes that return values:
    id ="string"
    dataframeCount ="string"
    dataframeName ="string"
>

    No Child Elements
</aims:iterateDataframe >
```

Bold: Attribute or child element is required.

Description:

Iterates through data frames in a Data Frames collection.

Restrictions:

None

Notes:

None

Attribute descriptions for iterateDataframe:

Attribute	Usage
dataframeCollection	Data Frames Collection object passed from aims:arcMapService *id*.
dataframeCount	Returns the number of data frames in the layout.
dataframeName	Returns the data frame name for the Map object.
id	Returns the ArcIMS Map object.

Examples for iterateDataframe:

Example 1:

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>iterateDataframe Sample</title>
</head>
<body>
<aims:tcpConnection id="myConnection" host="localhost" port="5300" debug="true" />

<aims:arcMapService id="arcmyDataframeCollection" connectionId="<%=(myConnection)%>"
name="arcMapService" error="arcmyError" />

<aims:getArcMapLayout dataframeCollection="<%=(arcmyDataframeCollection)%>" width="500"
height="700">
  <aims:envelope minx="0" miny="0" maxx="50" maxy="40" />
  <aims:layoutOutput type="jpg" />
  <aims:iterateDataframe id="myMapObject" dataframeName="dfn" dataframeCount="dfc"
dataframeCollection="<%=(arcmyDataframeCollection)%>">
  Dataframe Name : <%=(dfn)%><br>
  Dataframe Count : <%=(dfc)%><br>
<%
  if (dfn.equals("South East Asia")){
%>
```

```
    <aims:dataframe dataframeName="<%=(dfn)%>">
      <aims:featureCoordSys id="102030" />
      <aims:filterCoordSys id="102030" />
      <aims:scale rf="3251279" x="-5079159.83" y="2146309.62"  />
    </aims:dataframe>
<%
  }
%>
  </aims:iterateDataframe>
</aims:getArcMapLayout>

<aims:getArcMapLayoutAttribute id="arclayoutURL" value="LAYOUTURL"
dataframeCollection="<%=(arcmyDataframeCollection)%>" />
<img src="<%=(arclayoutURL)%>" /><br>

</body>
</html>
```

iterateEnvelope

Used in: mapping
Parent elements: aims:recordset

```
<aims:iterateEnvelope
    Attributes that return values:
    maxx ="double"
    maxy ="double"
    minx ="double"
    miny ="double"
>

    No Child Elements
</aims:iterateEnvelope >
```

Bold: Attribute or child element is required.

Description:

Iterates through envelopes in a recordset.

Restrictions:

None

Notes:

None

Attribute descriptions for iterateEnvelope:

Attribute	Usage
maxx	Maximum x-coordinate in map units.
maxy	Maximum y-coordinate in map units.
minx	Minimum x-coordinate in map units.
miny	Minimum y-coordinate in map units.

Examples for iterateEnvelope:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName" />

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:layer layerId="6" visible="true">
    <aims:filter envelope="true" whereExpression="NAME='San Jose'" />
    <aims:recordset>
      <b>Envelope </b><br>
      <aims:iterateEnvelope maxx="maxx" maxy="maxy" minx="minx" miny="miny" >
        Max X Value   : <%=(maxx)%> <br>
        Max Y Value   : <%=(maxy)%> <br>
        Min X Value   : <%=(minx)%> <br>
        Min Y Value   : <%=(miny)%> <br>
      </aims:iterateEnvelope>
    </aims:recordset>
  </aims:layer>
</aims:map>
```

iterateGeometry

Used in: mapping
Parent elements: aims:recordset

```
<aims:iterateGeometry
    Attributes that return values:
    x ="double"
    y ="double"
    recordNumber ="integer"
>

    No Child Elements
</aims:iterateGeometry >
```

Bold: Attribute or child element is required.

Description:

Iterates through the geometry of a feature in a recordset.

Restrictions:

None

Notes:

None

Attribute descriptions for iterateGeometry:

Attribute	Usage
recordNumber	Returns the record position of the Geometry object.
x	Returns the x-coordinate of the Points object.
y	Returns the y-coordinate of the Points object.

Examples for iterateGeometry:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName"/>

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:layer layerId="6" visible="true">
    <aims:filter geometry="true" whereExpression="NAME='San Jose'" />
    <aims:recordset>
      <b>Geometry  </b><br>
      <aims:iterateGeometry x="myX" y="myY">
        X Value   : <%=(myX)%>
        Y Value   : <%=(myY)%> <br>
      </aims:iterateGeometry>
    </aims:recordset>
  </aims:layer>
</aims:map>
```

iterateLayers

Used in: mapping
Parent elements: aims:map

<aims:iterateLayers

Attributes that set values:
layerId ="*string*"
reverse ="true | false" **[false]**
type ="ACETATE | FEATURE | IMAGE | ALL" **[ALL]**

Attributes that return values:
count ="*integer*"
>

(m) <getLayerAttribute... />

</aims:iterateLayers >

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Iterates through the Layers collection returned by aims:map.

Restrictions:

None

Notes:

None

Attribute descriptions for iterateLayers:

Attribute	Usage
count	Returns the count for the specified type.
layerId	Specifies which layer to iterate.
reverse	When set to "false", the order of the layers is the same as in the map configuration file. When set to "true", the layer order is reversed.
type	Sets the type of layers to iterate.

Examples for iterateLayers:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />
<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="serviceName"
/>
<aims:map id="myMap" serviceId="<%=myMapService%>" width="300" height="100">
<aims:iterateLayers count="layerCount">
  <aims:getLayerAttribute id="layerId" attribute="ID" />
    ID : <%=layerId%>
  <aims:getLayerAttribute id="layerName" attribute="NAME" />
    Layer Name : <%=layerName%><br>
</aims:iterateLayers>
Layer Count : <%=(layerCount)%>
</aims:map>
```

iterateMapService

Used in: **mapping** **metadata**
Parent elements: None

```
<aims:iterateMapService
    Attributes that set values:
    host ="string"
    password ="string"
    port ="string"
    url ="string"
    username ="string"

    Attributes that return values:
    name ="string"
    access ="string"
    count ="integer"
    serviceGroup ="string"
    status ="string"
    type ="string"
>

    No Child Elements
</aims:iterateMapService >
```

Bold: Attribute or child element is required.

Description:

Iterates through ArcIMS services based on the host name specified.

Restrictions:

- One of the following combinations should be included:
  - *host* and *port*
  - *url*

Notes:

None

Attribute descriptions for iterateMapService:

Attribute	Usage
access	Access type of the Virtual Server.
count	Total number of services in the service group.
host	The ArcIMS Application Server host name.
name	Name of the service.
password	Password when service authentication is used.
port	The ArcIMS Application Server port.
serviceGroup	Service group to which the service belongs.
status	Status of the service.
type	Type of service.
url	The URL of the machine such as http://mymachine.domain.com.
username	Username when service authentication is used.

Examples for iterateMapService:

Example 1: When using url.

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>iterateMapService</title>
</head>
<body>
<aims:iterateMapService url="http://localhost" username="publish" password="publish"
name="serviceName" access="serviceAccess" count="serviceCount" >
Service Name : <%=(serviceName)%><br>
```

```
Access       : <%=(serviceAccess)%><br>
Count        : <%=(serviceCount)%><br>
</aims:iterateMapService>
</body>
</html>
```

Example 2: When using host and port.

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>iterateMapService</title>
</head>
<body>
<aims:iterateMapService host="localhost" port="5300" name="serviceName"
access="serviceAccess" serviceGroup="serviceGroup" status="serviceStatus"
type="serviceType" count="serviceCount" >
Service Name : <%=(serviceName)%><br>
Access       : <%=(serviceAccess)%><br>
Service Group: <%=(serviceGroup)%><br>
Status       : <%=(serviceStatus)%><br>
Type         : <%=(serviceType)%><br>
Count        : <%=(serviceCount)%><br>
</aims:iterateMapService>
</body>
</html>
```

iterateMetadata

Used in: metadata
Parent elements: None

```
<aims:iterateMetadata
    Attributes that pass objects:
    iterate ="object"

    Attributes that return values:
    size ="string"
>

    (m) <datasetProperty... />

</aims:iterateMetadata >
```

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Iterates through the Metadata Document collection returned by aims:browse or aims:search.

Restrictions:

None

Notes:

None

Attribute descriptions for iterateMetadata:

Attribute	Usage
iterate	The Document collection returned by aims:browse or aims:search.
size	Returns the number of items in the document collection.

Examples for iterateMetadata:

Example 1: Iterates through the Metadata Document collection returned by aims:browse or aims:search.

```
<%-- Get Server Settings --%>
<aims:getSettings id="hostName" value="meta_host_name" />
<aims:getSettings id="portNumber" value="meta_port_number" />
<aims:getSettings id="serviceName" value="meta_service_name" />

<%-- Get Connection --%>
<aims:tcpConnection id="myConnection" host="<%= hostName %>" port="<%= portNumber %>"
debug="true" />

<% String docId = request.getParameter("docId"); %>

<aims:browse
      id="browseChildren"
      connectionId="<%= myConnection %>"
      service="<%= serviceName %>"
      docId="<%= docId %>"
      folderMask="2" />

<aims:iterateMetadata iterate="<%= browseChildren %>" size="mySize">

      <% if (!printedHeader) { %>
            <span class="explorerText">Categories</span>
            <ul>
            <% printedHeader = true; %>
      <% } %>

      <aims:datasetProperty id="theName" value="NAME" />
      <aims:datasetProperty id="theDocId" value="DOCID" />

      <li><a href="JavaScript:browseFolder('<%= URLEncoder.encode(theName) %>','<%=
URLEncoder.encode(theDocId) %>');"><%= theName %></a></li>

</aims:iterateMetadata>
```

iterateRasterInfo

Used in: **mapping**
Parent elements: None

```
<aims:iterateRasterInfo
    Attributes that pass objects:
    rasterInfo ="object"

    Attributes that return values:
    rasterId ="string"
    number ="string"
    value ="string"
>

    No Child Elements
</aims:iterateRasterInfo >
```

Bold: Attribute or child element is required.

Description:

Iterates through the raster information of a layer.

Restrictions:

None

Notes:

None

Attribute descriptions for iterateRasterInfo:

Attribute	Usage
number	Returns the Band Number.
rasterId	Returns the Raster ID.
rasterInfo	RasterInfo object passed from aims:getRasterInfo *id*.
value	Returns the Band Value.

Examples for iterateRasterInfo:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceImage"/>

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">

  <aims:getRasterInfo id="myRaster" layerId="0" x="1099.067" y="1349.42" />

  <aims:iterateRasterInfo rasterInfo="<%=(myRaster)%>" rasterId="id" number="no"
value="val">
    <b>Raster Id = </b> <%=(id)%>
    <b>Number    = </b> <%=(no)%>
    <b>Value     = </b> <%=(val)%><br>
  </aims:iterateRasterInfo>
</aims:map>
```

iterateRecordset

Used in: mapping
Parent elements: aims:recordset

```
<aims:iterateRecordset
    Attributes that return values:
    fieldValue ="string"
    recordLength ="integer"
    recordsetCount ="integer"
>

    No Child Elements
</aims:iterateRecordset >
```

Bold: Attribute or child element is required.

Description:

Iterates through recordset in the RecordSet collection.

Restrictions:

None

Notes:

None

Attribute descriptions for iterateRecordset:

Attribute	Usage
fieldValue	Returns the field value of the record.
recordLength	Returns the number of columns in a record.
recordsetCount	Returns the total number of records.

Examples for iterateRecordset:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName"/>

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:layer layerId="6" visible="true">
    <aims:filter whereExpression="NAME='San Jose'" />
    <aims:recordset>
      <b>Recordset</b><br>
      <aims:iterateRecordset  fieldValue="fValue">
        Field Value   : <%=(fValue)%> <br>
      </aims:iterateRecordset>
    </aims:recordset>
  </aims:layer>
</aims:map>
```

iterateServices

Used in: administration
Parent elements: None

```
<aims:iterateServices
    Attributes that pass objects:
    services ="object"

    Attributes that return values:
    count ="integer"
>

    (m) <getServiceAttribute... />

</aims:iterateServices >
```

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Iterates through a collection of services or an individual service to display information about a service.

Restrictions:

None

Notes:

None

Attribute descriptions for iterateServices:

Attribute	Usage
count	Returns the number of services iterated.
services	A Service or ServiceCollection object retrieved from aims:getService, aims:createService, or aims:getServices.

Examples for iterateServices:

Example 1:

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://mymachine"
      role="admin"
      username="admin"
      password="admin"
/>

<aims:getServices id="myServices" connectionId="<%=myhttpConnection%>" />

<aims:iterateServices services="<%=myServices%>" count="myCount" >
  <aims:getServiceAttribute id="theName" attribute="NAME" />
  <aims:getServiceAttribute id="theFile" attribute="CONFIGFILE" />
  <aims:getServiceAttribute id="theType" attribute="IMAGETYPE" />
  <aims:getServiceAttribute id="theOutDir" attribute="OUTPUTDIR" />
  <aims:getServiceAttribute id="theOutClean" attribute="OUTPUTCLEANUP" />
  <aims:getServiceAttribute id="theOutURL" attribute="OUTPUTURL" />
  <aims:getServiceAttribute id="thePixel" attribute="PIXELCOUNT" />
  <aims:getServiceAttribute id="theVSName" attribute="VIRTUALSERVERNAME" />
  <aims:getServiceAttribute id="theVSInfo" attribute="VIRTUALSERVERINFO" />
  <aims:getServiceAttribute id="theStatus" attribute="STATUS" />
  <aims:getServiceAttribute id="theDes" attribute="DESCRIPTION" />
  <aims:getServiceAttribute id="theAcc" attribute="ACCESS" />
  <td>
```

```
  <br><b>Iterate all services</b>
  <br><b>Service name: <%=theName%></b>
  <br><b>Config file: <%=theFile%></b>
  <br><b>Image Type: <%=theType%></b>
  <br><b>Pixelcount: <%=thePixel%></b>
  <br><b>Virtual Server Name: <%=theVSName%></b>
  <br><b>Virtual Server Info: <%=theVSInfo%></b>
  <br><b>Output Dir: <%=theOutDir%></b>
  <br><b>Output URL: <%=theOutURL%></b>
  <br><b>Output Cleanup: <%=theOutClean%></b>
  <br><b>Description: <%=theDes%></b>
  <br><b>Access: <%=theAcc%></b>
  <br><b>Status: <%=theStatus%></b>
  <br><b></b>
  </td>
</aims:iterateServices>
```

iterateTableDesc

Used in: mapping
Parent elements: aims:recordset

```
<aims:iterateTableDesc
    Attributes that return values:
    fieldName ="string"
    fieldLength ="integer"
    fieldPrecision ="double"
    fieldType ="string"
>

    No Child Elements
</aims:iterateTableDesc >
```

Bold: Attribute or child element is required.

Description:

Iterates through column names of a table.

Restrictions:

None

Notes:

None

Attribute descriptions for iterateTableDesc:

Attribute	Usage
fieldLength	Returns the length of the field for the recordset.
fieldName	Returns the name of the field for the recordset.
fieldPrecision	Returns the precision of the field for the recordset.
fieldType	Returns the type of the field for the recordset.

Examples for iterateTableDesc:

Example 1:

```
<aims:tcpConnection id="myTCPConnection"  host="myMachine" port="5300" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>"
name="<%=myService%>" loadRecordset="true" loadEnvelope="true" loadExtensions="true"
loadRenderer="true" />

<aims:map id="myMap" legend="myLegend" envelope="myEnvelope"
serviceId="<%=myMapService%>" width="500" height="300">
  <aims:separators  ts="," cs=" " />
  <aims:layer layerId="0" visible="true">
    <aims:filter envelope="true" geometry="true" globalEnvelope="true">
      <aims:line coords="-122.483 37.769,-122.482 37.763" />
    </aims:filter>
    <aims:recordset>
      <aims:iterateTableDesc fieldName="fName" fieldType="fType" fieldPrecision="fPrec"
fieldLength="fLen">
        <%=(fName)%>
        <%=(fType)%>
        <%=(fPrec)%>
        <%=(fLen)%> <br>
      </aims:iterateTableDesc>
    </aims:recordset>
  </aims:layer>
</aims:map>
```

iterateVirtualServers

Used in: administration
Parent elements: None

```
<aims:iterateVirtualServers

    Attributes that pass objects:
    virtualServers ="object"

    Attributes that return values:
    count ="integer"
>

    (m) <getVirtualServerAttribute... />

</aims:iterateVirtualServers >
```

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Iterates through a collection of Virtual Servers or an individual Virtual Server to display information about a Virtual Server.

Restrictions:

None

Notes:

None

Attribute descriptions for iterateVirtualServers:

Attribute	Usage
count	Returns the number of Virtual Servers iterated.
virtualServers	A Virtual Server collection retrieved from aims:getVirtualServer, aims:createVirtualServer, or aims:getVirtualServers.

Examples for iterateVirtualServers:

Example 1:

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://mymachine"
      role="admin"
      debug="true"
      username="admin"
      password="admin"
/>

<aims:getVirtualServers id="myServers" connectionId="<%=myhttpConnection%>"
error="getError" />

<aims:iterateVirtualServers virtualServers="<%=myServers%>" count="serverCount" >
      <aims:getVirtualServerAttribute id="name" attribute="NAME" />
      <aims:getVirtualServerAttribute id="type" attribute="TYPE" />
      <aims:getVirtualServerAttribute id="access" attribute="ACCESS" />
      <aims:getVirtualServerAttribute id="description" attribute="DESCRIPTION" />
      <aims:getVirtualServerAttribute id="version" attribute="VERSION" />
      <aims:getVirtualServerAttribute id="containerset" attribute="CONTAINERSET" />
      <td>
        <br><b>name is: <%=name%></b>
        type is: <%=type%>
        description is: <%=description%>
        version is: <%=version%>
        access is: <%=access%>
        containerset is: <%=containerset%>
      </td>
</aims:iterateVirtualServers>
```

layer

Used in: mapping
Parent elements: aims:dataframe aims:map

<aims:layer

Attributes that set values:
layerId ="*string*"
extractable ="true | false" **[false]**
visible ="true | false"
>

Nested tag for showing attribute information:
(m) <recordSet... />

Nested tags for defining a renderer:
<groupRenderer... /> *[Or]*
<scaleDependentRenderer... /> *[Or]*
<simpleLabelRenderer... /> *[Or]*
<simpleRenderer... /> *[Or]*
<valueMapLabelRenderer... /> *[Or]*
<valueMapRenderer... /> *[Or]*

Nested tags for retrieving layer extent:
<getEnvelope... />

Nested tags for showing a subset of data:
<displayFeatures... />
<filter... />

Nested tags used for geocoding:
<addressMatchInputs... />
<iterateAddressMatchInputs... />
<iterateAddressMatchResults... />

Nested tags when aims:dataframe is parent tag:
No Child Elements

</aims:layer >

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Changes attributes of a specified layer.

Restrictions:

None

Notes:

None

Attribute descriptions for layer:

Attribute	Usage
extractable	Determines whether layer can be extracted when aims:map *extract*="true".
layerId	Layer ID defined in a map configuration file or from layer added using aims:createLayer.
visible	Toggles the visibility of the layer.

Examples for layer:

Example 1:

```
<%@page contentType="text/html"%>
<%@taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>displayFeatures with whereExpression</title>
</head>
<body>
<aims:tcpConnection id="myTCPConnection" host="localhost" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="SantaClara"
/>

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">

<aims:layer layerId="5" visible="true">
  <aims:displayFeatures whereExpression="NAME='San Jose'" zoomToFeatures="false">
    <aims:simpleMarkerSymbol width="25" type="star" color="0,0,255" outline="0,0,0"
shadow="0,0,0" />
  </aims:displayFeatures>
</aims:layer>

</aims:map>
<img src="<%=(myMap)%>" />
</body>
</html>
```

layerList

Used in: mapping
Parent elements: aims:dataframe

```
<aims:layerList
    Attributes that set values:
    order ="true | false"  [true]
>

    No Child Elements
</aims:layerList >
```

Description:

Defines list of layers to be drawn in a layout data frame.

Restrictions:

None

Notes:

When using ArcXML, the default for LAYERLIST *order* is "false". When using the Java Connector, the default for *order* is "true".

Attribute descriptions for layerList:

Attribute	Usage
order	When set to "true", only layers in the LAYERLIST are drawn. All layers, including acetate layers, must have a unique ID.

Examples for layerList:

Example 1: When used in a layout.

```
<aims:tcpConnection id="myConnection" host="localhost" port="5300" debug="true" />

<aims:arcMapService id="arcmyDataframeCollection" connectionId="<%=(myConnection)%>"
name="multidf" error="arcmyError" />

<aims:getArcMapLayout dataframeCollection="<%=(arcmyDataframeCollection)%>" width="500"
height="700">
  <aims:envelope minx="0" miny="0" maxx="50" maxy="40" />
  <aims:layoutOutput type="jpg" />
  <aims:dataframe dataframeName="Europe">
    <aims:layerList order="true"/>
    <aims:layer layerId="1" visible="false" />
    <aims:layer layerId="0" visible="false" />
    <aims:envelope minx="19.8889685482977" miny="52.7973466116867"
maxx="35.9893558857411" maxy="63.792074206216" />
  </aims:dataframe>
</aims:getArcMapLayout>

<aims:getArcMapLayoutAttribute id="arclayoutURL" value="LAYOUTURL"
dataframeCollection="<%=(arcmyDataframeCollection)%>" />
<img src="<%=(arclayoutURL)%>" /><br>

</body>
</html>
```

layoutOutput

Used in: mapping
Parent elements: None

```
<aims:layoutOutput
    Attributes that set values:
    baseURL ="string"
    name ="string"
    path ="string"
    type ="ai | bmp | emf | eps | jpg | pdf | png8 | png24"  [jpg]
    url ="string"
>

    No Child Elements
</aims:layoutOutput >
```

Description:

Defines a pathname and URL for ArcMap layouts.

Restrictions:

None

Notes:

None

Attribute descriptions for layoutOutput:

Attribute	Usage
baseURL	Paired with *path*. URL of output directory if default filename is generated by ArcIMS. Do not include a filename.
name	Paired with *url*. User assigns an output filename. Use full pathname along with the filename. Only filenames with *.ai, *.bmp, *.emf, *.eps, *.jpg, *.pdf, *.png8, or *.png24 extensions are valid. UNC pathnames are valid (\\myComputer\arcims\output\myfile.jpg).
path	Paired with *baseURL*. Directory to output file generated by Image Server. Do not include the filename. UNC pathnames are valid (\\myComputer\arcims\output).
type	Output layout file type.
url	Paired with *name*. URL of output file. Include filename as part of URL. The filename must match the filename used in *name*.

Examples for layoutOutput:

Example 1: When using name and url attribute pair.

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>layoutOutput tag example</title>
</head>
<body>
<aims:tcpConnection id="myConnection" host="localhost" port="5300" debug="true" />

<aims:arcMapService id="arcmyDataframeCollection" connectionId="<%=(myConnection)%>"
name="arcMapService" error="arcmyError" />

<aims:getArcMapLayout dataframeCollection="<%=(arcmyDataframeCollection)%>" width="500"
height="700">
  <aims:envelope minx="0" miny="0" maxx="50" maxy="40" />
  <aims:layoutOutput name="C:/ArcIMS/Output/TestLayout.pdf"
url="http://localhost/Output/TestLayout.pdf" type="pdf" />
```

```
</aims:getArcMapLayout>

<aims:getArcMapLayoutAttribute id="arclayoutURL" value="LAYOUTURL"
dataframeCollection="<%=(arcmyDataframeCollection)%>" />
<img src="<%=(arclayoutURL)%>" /><br>

</body>
</html>
```

Example 2: When using path and baseURL attribute pair.

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>layoutOutput tag example</title>
</head>
<body>
<aims:tcpConnection id="myConnection" host="localhost" port="5300" debug="true" />

<aims:arcMapService id="arcmyDataframeCollection" connectionId="<%=(myConnection)%>"
name="arcMapService" error="arcmyError" />

<aims:getArcMapLayout dataframeCollection="<%=(arcmyDataframeCollection)%>" width="500"
height="700">
  <aims:envelope minx="0" miny="0" maxx="50" maxy="40" />
  <aims:layoutOutput path="C:/ArcIMS/Output" baseURL="http://localhost/Output" type="jpg"
/>
</aims:getArcMapLayout>

<aims:getArcMapLayoutAttribute id="arclayoutURL" value="LAYOUTURL"
dataframeCollection="<%=(arcmyDataframeCollection)%>" />
<img src="<%=(arclayoutURL)%>" /><br>

</body>
</html>
```

legend

Used in: mapping
Parent elements: aims:map

<aims:legend

Attributes that set values:

PROPERTIES when using ArcMap Server:
autoExtend ="true | false" **[false]**
display ="true | false" **[true]**
font ="Any system font" **[Arial]**
height ="1 - NNN" **[300]**
swatchHeight ="*integer*" **[14]**
swatchWidth ="*integer*" **[18]**
title ="*string*"
titleFontSize ="*integer*" **[12]**
width ="1 - NNN" **[125]**

When using Image Server:
antialiasing ="true | false" **[true]**
autoExtend ="true | false" **[false]**
backgroundColor ="0,0,0 - 255,255,255"
canSplit ="true | false" **[false]**
cellSpacing ="*integer*" **[2]**
columns ="*integer*" **[1]**
display ="true | false" **[true]**
font ="Any system font" **[Arial]**
height ="1 - NNN" **[300]**
layerFontSize ="*integer*" **[10]**
reverseOrder ="true | false" **[false]**
splitText ="*string*" **[(cont)]**
swatchHeight ="*integer*" **[14]**
swatchWidth ="*integer*" **[18]**
title ="*string*"

```
    titleFontSize ="integer"  [12]
    transColor ="0,0,0 - 255,255,255"
    valueFontSize ="integer"  [8]
    width ="1 - NNN"  [125]
>

    (m) <hideLayer... />

</aims:legend >
```
(m): Child element can be used multiple times.

Description:

Defines the legend attributes of the map.

Restrictions:

None

Notes:

None

Attribute descriptions for legend:

Attribute	Usage
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
autoExtend	If "true", automatically extends legend vertically past size specified in height, if needed.
backgroundColor	Legend's background color using RGB values.
canSplit	Allows splitting of valuemap layers between columns.
cellSpacing	Defines number of pixels to pad between entries.
columns	Defines number of columns in legend.
display	Turns legend on or off.
font	Font for title. The name is case sensitive. If font name uses "&", use "&" instead. For example, ESRI Transportation & Civic should be written as ESRI Transportation & Civic.
height	Legend height in pixels.
layerFontSize	Font size for layer labels.
reverseOrder	Reverse order of layers.
splitText	Text that displays in bottom of a column if a valuemap is carried over into the next column.
swatchHeight	Height of the symbol swatch in pixels.
swatchWidth	Width of the symbol swatch in pixels.
title	Title of legend.
titleFontSize	Font size for title.
transColor	Transparency color using RGB values. White is 255,255,255.
valueFontSize	Font size for value map labels.
width	Legend width in pixels.

Examples for legend:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" debug="true"/>
<aims:mapService id="myService" connectionId="<%=myTCPConnection%>" name="serviceName" />
<aims:map id="myMapURL" legend="myLegend" serviceId="<%=myService%>">
  <aims:legend title="Legend" font="Arial" autoExtend="true" columns="1" width="170"
height="300" backgroundColor="255,255,0" />
</aims:map>
<img src="<%=(myMapURL)%>" />
<img src="<%=(myLegend)%>" />
```

legendOutput

Used in: mapping
Parent elements: aims:map

```
<aims:legendOutput
    Attributes that set values:
    legendBaseURL ="string"
    legendName ="string"
    legendPath ="string"
    legendURL ="string"
    type ="gif | jpg | png | png8"  [jpg]
>

    No Child Elements
</aims:legendOutput >
```

Description:

Defines a pathname and URL for legend output images.

Restrictions:

LegendOutput works with paired attributes. If one of the attributes is used, its pair is also required. The attribute pairs are listed in the table below.

Attribute	Paired Attribute	Filename Assignment
legendPath	legendBaseURL	ArcIMS assigns random filename.
legendName	legendURL	User assigns a filename.

Notes:

- If aims:mapOutput *type* and aims:legendOutput *type* have different values, then aims:mapOutput takes precedence.

- If aims:legendOutput is used, the output files are not automatically deleted by ArcIMS Tasker. In order for the files to be deleted, the *taskfile* property must be set in tasker.properties. For information on setting this property, see the ArcIMS Help.

Attribute descriptions for legendOutput:

Attribute	Usage
legendBaseURL	Paired with *legendPath*. URL of output directory if default legend filename is generated by ArcIMS. Do not include a filename.
legendName	Paired with *legendURL*. User assigns an output legend filename. Use full pathname along with the filename. The filename must match the filename used in *legendURL*. UNC pathnames are valid (\\myComputer\arcims\output\mylegend.jpg).
legendPath	Paired with *legendBaseURL*. Directory to output legend file generated by Image Server. Do not include the filename. UNC pathnames are valid (\\myComputer\arcims\output).
legendURL	Paired with *legendName*. URL of output legend file. Include filename as part of URL. The filename must match the filename used in *legendName*.
type	Output image file type.

Examples for legendOutput:

Example 1: When using legendName and legendURL attribute pair.

```
<%@page contentType="text/html"%>
<%@taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>mapOutput and legendOutput tag example</title>
</head>
<body>
<aims:tcpConnection id="myTCPConnection" host="localhost" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="SantaClara"
/>

<aims:map id="myMap" legend="myLegend" serviceId="<%=(myMapService)%>" width="500"
height="300">
```

```
  <aims:mapOutput name="C:/ArcIMS/Output/TestMap.png"
url="http://localhost/output/TestMap.png" type="png" />
  <aims:legendOutput legendName="C:/ArcIMS/Output/TestLegend.png"
legendURL="http://localhost/output/TestLegend.png" />
</aims:map>

<img src="<%=(myMap)%>" />
<img src="<%=(myLegend)%>" />
</body>
</html>
```

Example 2: When using legendPath and legendBaseURL attribute pair.

```
<%@page contentType="text/html"%>
<%@taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>mapOutput and legendOutput tag example</title>
</head>
<body>
<aims:tcpConnection id="myTCPConnection" host="localhost" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="SantaClara"
/>

<aims:map id="myMap" legend="myLegend" serviceId="<%=(myMapService)%>" width="500"
height="300">
  <aims:mapOutput path="C:/ArcIMS/Output" baseURL="http://localhost/output" type="png" />
  <aims:legendOutput legendPath="C:/ArcIMS/Output"
legendBaseURL="http://localhost/output" />
</aims:map>

<img src="<%=(myMap)%>" />
<img src="<%=(myLegend)%>" />
</body>
</html>
```

line

Used in: mapping
Parent elements: aims:acetateObject aims:displayFeatures aims:filter

```
<aims:line
    Attributes that set values:
    coords ="x1 y1;...xn yn"
    lineId ="string"
    name ="string"
>

    When parent element is aims:acetateObject:
    <hashLineSymbol... />  [Or]
    <rasterMarkerSymbol... />  [Or]
    <simpleLineSymbol... />  [Or]
    <simpleMarkerSymbol... />  [Or]
    <trueTypeMarkerSymbol... />  [Or]

    When parent element is aims:filter or aims:displayFeatures:
        No Child Elements

</aims:line >
```

Bold: Attribute or child element is required.

Description:

Draws a line as an acetate layer on the map or is used as a filter to select features.

Restrictions:

- Line must be continuous with no breaks.
- Symbols can be nested only in an object, not in a filter.

Notes:

None

Attribute descriptions for line:

Attribute	Usage
coords	X,y coordinates representing a line. Coordinate x,y values are separated by white space, and coordinate pairs are separated by a semicolon by default. The separators can be changed by using aims:separators.
lineId	Identifier for the line.
name	Name of the line.

Examples for line:

Example 1:

```
<aims:tcpConnection id="myTCPConnection"  host="myMachine" port="5300" />
<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="serviceName"
/>
<aims:map id="myMap" serviceId="<%=myMapService%>" width="300" height="500">
  <aims:createLayer layerId="lineLayer" type="acetate" name="roads" visible="true" >
    <aims:acetateObject units="pixel">
      <aims:line coords="100 150;300 150">
        <aims:simpleLineSymbol width="10" type="dash" transparency=".5" />
      </aims:line>
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>
<img src="<%=(myMap)%>" />
```

loadElements

Used in: metadata
Parent elements: None

```
<aims:loadElements
    Attributes that set values:
    names ="string"
    url ="string"

    Attributes that return values:
    id ="string"
>

    No Child Elements
</aims:loadElements >
```

Bold: Attribute or child element is required.

Description:

Retrieves a value from a metadata document for each xpath.

Restrictions:

None

Notes:

None

Attribute descriptions for loadElements:

Attribute	Usage
id	Returns the elements for access by aims:getElement.
names	A list of element names to load. Each name must be defined by the xpath or location of an element inside an XML tree. For example, the element "title" might be inside an XML structure such as <idinfo> <citation> <citeinfo> **<title>...</title>** </citeinfo> </citation> </idinfo> To define "title", the string value for this attribute would be "idinfo/citation/citeinfo/**title**". Values are separated in the list by a comma.
url	The URL to the metadata document.

Examples for loadElements:

Example 1:

```
<%
String names = "metadata/idinfo/citation/citeinfo/title,"
      + "metadata/Esri/gn/contenttype,"
      + "metadata/Esri/gn/coverage,"
      + "metadata/spdoinfo/direct,"
      + "metadata/idinfo/browse/browsen,"
      + "metadata/idinfo/browse/browsed,"
      + "metadata/idinfo/citation/citeinfo/pubinfo/publish,"
      + "metadata/dataqual/lineage/srcinfo/srcscale,"
      + "metadata/Esri/gn/resolution";
%>
```

```
<aims:loadElements id="theElements" url="<%= metadataUrl %>" names="<%= names %>" />

<aims:getElement id="title" loadedElements="<%= theElements %>"
value="metadata/idinfo/citation/citeinfo/title" />
```

map

Used in: **mapping**
Parent elements: None

<aims:map

Attributes that set values:
autoresize ="true | false" **[false]**
background ="0,0,0 - 255,255,255"
bufferImage ="true | false" **[false]**
bufferRegion ="true | false" **[false]**
createMap ="true | false" **[true]**
dpi ="1 - NNN"
drawMap ="true | false" **[true]**
extract ="true | false" **[false]**
height ="*integer*" **[350]**
reset ="true | false" **[false]**
transcolor ="0,0,0 - 255,255,255"
width ="*integer*" **[400]**

Attributes that pass objects:
serviceId ="*object*"

Attributes that return values:
id ="*string*"
envelope ="*object*"
legend ="*string*"
legendStream ="*byte*"
legendStreamType ="JPG | GIF | PNG | PNG8"
stream ="*byte*"
streamType ="JPG | GIF | PNG | PNG8"
>

Nested tags for adding dynamic and acetate layers:
(m) <createLayer... />

Nested tags for identifying raster values:
<getRasterInfo... />

Nested tags for layer order:
<moveLayer... />
<moveLayerToBottom... />
<moveLayerToTop... />

Nested tags for map legend:
<legend... />

Nested tags for modifying layers in a map configuration file:
<layer... />

Nested tags for navigation:
<envelope... />
<pan... />
<zoom... />
<zoomFullExtent... />

Nested tags for object handling:
<getLayerId... />
(m) <iterateLayers... />
<removeLayer... />

Nested tags for output:
<extractOutput... />
<legendOutput... />
<mapOutput... />

Nested tags for projections:
<featureCoordSys... />
<filterCoordSys... />

Nested tags for retrieving current extent:
<getEnvelope... />

Nested tags for separator information:
<separators... />

</aims:map >

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Creates a map image.

Restrictions:

- If the attribute *drawMap* is set to "false", then the *legend* attribute is required.
- When using the image stream attributes *stream*, *streamType*, *legendStream*, and *legendStreamType*, you must have the following line in the map configuration file inside the PROPERTIES section:

  <OUTPUT method="stream"/>
- Image streaming is valid only with Image Services. It is not valid with ArcMap Image Services.
- When using *dpi*, you must also include *height* and *width*.

Notes:

In order to stream images, two JSP files are needed:

1. The main JSP file that writes out the HTML page, such as Example 2 below.
2. A second JSP file that processes the stream, such as Example 3 below.

Attribute descriptions for map:

Attribute	Usage
autoresize	The maximum generated image size is based on the image memory limit set when an Image Service is started. For example, an image memory limit of 1 MB allows a map no larger than 262,144 pixels (512 x 512) to be generated. If *autoresize* is set to "true", a requested map greater than the maximum pixel count will be reduced in size to within the maximum pixel count. If *autoresize* is set to "false", no image is generated and an error message is returned by the ArcIMS Spatial Server.
background	Background color using RGB values defined as a comma-delimited string.

Attribute	Usage
bufferImage	Highlights the buffered features.
bufferRegion	Draws the buffered region on the map.
createMap	When set to "false", the map image is not returned. If a legend image is requested, only the legend image is returned.
dpi	Dots per inch (dpi).
drawMap	Turns map generation off when only a legend is needed.
envelope	Returns Envelope object of generated map.
extract	Extracts layers based on the current extent in the viewer. When set to "true", the request is sent to the Extract Server instead of the Image Server.
height	Height of the map image in pixels.
id	Returns map URL string. If *extract* is set to "true", then the extract URL string is returned.
legend	Returns URL string of the map's legend.
legendStream	Returns the legend image in Base64 encoded format.
legendStreamType	Returns the legend image format. If the image is 24-bit PNG, the returned value is "PNG". An 8-bit PNG file returns the value "PNG8".
reset	Resets the map back to its original state retrieved using aims:mapService.
serviceId	The name of the ArcIMS service.
stream	Returns the image in Base64 encoded format.
streamType	Returns the image format. If the image is 24-bit PNG, the returned value is "PNG". An 8-bit PNG file returns the value "PNG8".
transcolor	Transparent color using RGB values defined as a comma-delimited string.
width	Width of the map image in pixels.

Examples for map:

Example 1:

```
<aims:tcpConnection id="myTCPConnection"  host="myMachine" port="5300" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>"
```

```
      name="<%=myService%>" />

<aims:map id="myMap" legend="myLegend" envelope="myEnvelope"
serviceId="<%=(myMapService)%>" width="300" height="100" background="100,140,230"
transcolor="0,0,0" bufferImage="true" bufferRegion="true">

</aims:map>

<img src="<%=myMap%>" />
<img src="<%=myLegend%>" />
```

Example 2: When using image streaming. Must be used together with the following example (Example 3), which should be a separate file named "streamimage.jsp".

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>Streaming Samples</title>
<body bgcolor="#ffffff">
<center>
<%
java.util.Vector layer = new java.util.Vector();
%>
<aims:getSettings id="myHost" value="host" propFileName="jspsamples.properties" />
<aims:getSettings id="myPort" value="port" propFileName="jspsamples.properties" />
<aims:getSettings id="myService" value="streamservice"
propFileName="jspsamples.properties" />

<aims:tcpConnection id="myConnection" host="<%=(myHost)%>" port="<%=(myPort)%>"/>
<aims:mapService id="myMapService" connectionId="<%=(myConnection)%>"
name="<%=(myService)%>" />
<aims:map id="myMap" stream="myStream" streamType="myStreamType"
legendStream="myLegendStream" legendStreamType="myLegendStreamType"
serviceId="<%=(myMapService)%>" width="500" height="350">
  <aims:legend title="My Map" width="200" height="150" />
    <aims:iterateLayers count="count" type="feature">
```

```
        <aims:getLayerAttribute id="layerName" attribute="NAME" />
        <%layer.add(layerName);%>
    </aims:iterateLayers>
</aims:map>
<%
session.setAttribute("streamMap",myStream);
session.setAttribute("streamLegend",myLegendStream);
String mapSrc = "/src/streamimage.jsp?map=1&streamType=" + myStreamType;
String legendSrc = "/src/streamimage.jsp?map=2&streamType=" + myLegendStreamType;
%>
<table border="1" align="center">
<tr><td align="center"><img src="<%=(mapSrc)%>"></td>
<td align="left"><img src="<%=legendSrc%>"><br>
<b>List of Layers</b>
<%
for (int i=0;i<layer.size();i++)
      out.println(layer.get(i) + "<br>");
%>
<br>
</td></tr></table>

</center>
</body>
</html>
```

Example 3: When using image streaming. This example processes the stream. It should be saved as a file named "streamimage.jsp" and used together with Example 2.

```
<%
byte [] stream = null;
if (request.getParameter("map").equals("1")){
      if (session.getAttribute("streamMap") != null)
         stream = (byte [])session.getAttribute("streamMap");
}else{
      if (session.getAttribute("streamLegend") != null)
            stream = (byte [])session.getAttribute("streamLegend");
```

```
}

String  streamType = request.getParameter("streamType");
javax.servlet.ServletOutputStream outServ =  response.getOutputStream();

String contentType = "image/jpeg";

if (streamType.trim().length() != 0)
      contentType= "image/" + streamType.toLowerCase();
response.flushBuffer();
response.setContentType(contentType);

if (stream != null){
      for (int i = 0; i <stream.length;i++)
            outServ.write(stream[i]);
}
%>
```

mapOutput

Used in: mapping
Parent elements: aims:map

```
<aims:mapOutput
    Attributes that set values:
    baseURL ="string"
    name ="string"
    path ="string"
    type ="gif | jpg | png | png8"  [jpg]
    url ="string"
>

    No Child Elements
</aims:mapOutput >
```

Description:

Defines a pathname and URL for output map images.

Restrictions:

- MapOutput works with paired attributes. If one of the attributes is used, its pair is also required. The attribute pairs are listed in the table below.

Attribute	Paired Attribute	Filename Assignment
path	baseURL	ArcIMS assigns random filename.
name	url	User assigns a filename.

- GIF format is not valid with ArcMap Image Services.

Notes:

- If aims:mapOutput *type* and aims:legendOutput *type* have different values, then aims:mapOutput takes precedence.
- If aims:mapOutput is used, the output files are not automatically deleted by ArcIMS Tasker. In order for the files to be deleted, the *taskfile* property must be set in tasker.properties. For information on setting this property, see the ArcIMS Help.

Attribute descriptions for mapOutput:

Attribute	Usage
baseURL	Paired with *path*. URL of output directory if default filename is generated by ArcIMS. Do not include a filename.
name	Paired with *url*. User assigns an output filename. Use full pathname along with the filename. Only filenames with a *.jpg, *.png, or *.gif extension are valid. The filename must match the filename used in *url*. UNC pathnames are valid (\\myComputer\arcims\output\myfile.jpg).
path	Paired with *baseURL*. Directory to output file generated by Image Server. Do not include the filename. UNC pathnames are valid (\\myComputer\arcims\output).
type	Output image file type.
url	Paired with *name*. URL of output file. Include filename as part of URL. The filename must match the filename used in *name*.

Examples for mapOutput:

Example 1: When using name and url attribute pair.

```
<%@page contentType="text/html"%>
<%@taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>mapOutput and legendOutput tag example</title>
</head>
<body>
<aims:tcpConnection id="myTCPConnection" host="localhost" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="SantaClara"
/>
```

```
<aims:map id="myMap" legend="myLegend" serviceId="<%=(myMapService)%>" width="500"
height="300">
  <aims:mapOutput name="C:/ArcIMS/Output/TestMap.png"
url="http://localhost/output/TestMap.png" type="png" />
  <aims:legendOutput legendName="C:/ArcIMS/Output/TestLegend.png"
legendURL="http://localhost/output/TestLegend.png" />
</aims:map>

<img src="<%=(myMap)%>" />
<img src="<%=(myLegend)%>" />
</body>
</html>
```

Example 2: When using path and baseURL attribute pair.

```
<%@page contentType="text/html"%>
<%@taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>mapOutput and legendOutput tag example</title>
</head>
<body>
<aims:tcpConnection id="myTCPConnection"  host="localhost" port="5300" debug="true" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="SantaClara"
/>

<aims:map id="myMap" legend="myLegend" serviceId="<%=(myMapService)%>" width="500"
height="300">
  <aims:mapOutput path="C:/ArcIMS/Output" baseURL="http://localhost/output" type="png" />
  <aims:legendOutput legendPath="C:/ArcIMS/Output"
legendBaseURL="http://localhost/output" />
</aims:map>

<img src="<%=(myMap)%>" />
<img src="<%=(myLegend)%>" />
</body>
</html>
```

mapService

Used in: **mapping**
Parent elements: None

```
<aims:mapService

    Attributes that set values:
    name ="string"
    dpi ="1 - NNN"
    loadEnvelope ="true | false"  [false]
    loadExtensions ="true | false"  [false]
    loadRecordset ="true | false"  [false]
    loadRenderer ="true | false"  [false]
    password ="string"  [null]
    username ="string"  [null]

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    id ="string"
    error ="string"
>

    No Child Elements
</aims:mapService >
```

Bold: Attribute or child element is required.

Description:

Reads initial values from an Image Service, such as envelopes, extensions, renderers, and database field names.

Restrictions:

None

Notes:

For reading initial values from an ArcMap Image Service, see arcMapService.

Attribute descriptions for mapService:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
dpi	Dots per inch. Used for calculating the correct scale thresholds for scale-dependent elements. The *dpi* value overrides the value used in a service.
error	Returns an error string if any occurred. Returns "Success" if no error.
id	Returns an ArcIMS service object.
loadEnvelope	Loads individual layer envelopes from service.
loadExtensions	Loads individual layer extensions from service.
loadRecordset	Loads individual layer recordsets from service.
loadRenderer	Loads individual layer renderers from service.
name	ArcIMS service name.
password	Password if authentication required.
username	Username if authentication required.

Examples for mapService:

Example 1:

```
<aims:tcpConnection id="myTCPConnection"  host="myMachine" port="5300" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="serviceName"
/>

<aims:map id="myMap" legend="myLegend" envelope="myEnvelope"
serviceId="<%=(myMapService)%>" width="300" height="100" background="100,140,230"
transcolor="0,0,0" bufferImage="true" bufferRegion="true" />

<img src="<%=myMap%>" />
<img src="<%=myLegend%>" />
```

metadataChild

Used in: metadata
Parent elements: aims:deleteMetadataRelationship aims:putMetadataRelationship

```
<aims:metadataChild
    Attributes that set values:
    docId ="string"
>
    No Child Elements
</aims:metadataChild >
```

Bold: Attribute or child element is required.

Description:

Identifies a child document that has relationship to a source or parent metadata document.

Restrictions:

None

Notes:

None

Attribute descriptions for metadataChild:

Attribute	Usage
docId	String used to uniquely identify a document. The client used to publish the metadata is responsible for creating the document ID. This ID is automatically assigned when using ArcCatalog. If another client is used, aims:getUUID can be used to request a valid ID. The format for an ID is the following: {HHHHHHHH-HHHH-HHHH-HHHH-HHHHHHHHHHHH} where H is a hexadecimal digit (0–9,a–f,A–F). The ID is limited to 38 characters.

Examples for metadataChild:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" username="publish"
password="publish" />

<aims:putMetadataRelationship connectionId="<%=(myConnection)%>" service="serviceName"
docId="{E6F7632B-04CE-4B50-A995-887DD144F4DC}">

  <aims:metadataChild docId="{9F582D73-7001-4A4B-A806-0FB8392E1F8F}" />
  <aims:metadataSibling docId="{C7F0DEBF-BCF6-48E7-847C-72C31A1DBCBD}" />

</aims:putMetadataRelationship>
```

metadataSibling

Used in: metadata
Parent elements: aims:deleteMetadataRelationship aims:putMetadataRelationship

```
<aims:metadataSibling
    Attributes that set values:
    docId ="string"
>
    No Child Elements
</aims:metadataSibling >
```

Bold: Attribute or child element is required.

Description:

Identifies a sibling, or related, document to a source metadata document.

Restrictions:

None

Notes:

None

Attribute descriptions for metadataSibling:

Attribute	Usage
docId	String used to uniquely identify a document. The client used to publish the metadata is responsible for creating the document ID. This ID is automatically assigned when using ArcCatalog. If another client is used, aims:getUUID can be used to request a valid ID. The format for an ID is the following: {HHHHHHHH-HHHH-HHHH-HHHH-HHHHHHHHHHHH} where H is a hexadecimal digit (0–9,a–f,A–F). The ID is limited to 38 characters.

Examples for metadataSibling:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" username="publish"
password="publish" />

<aims:putMetadataRelationship connectionId="<%=(myConnection)%>" service="serviceName"
docId="{E6F7632B-04CE-4B50-A995-887DD144F4DC}">

  <aims:metadataChild docId="{9F582D73-7001-4A4B-A806-0FB8392E1F8F}" />
  <aims:metadataSibling docId="{C7F0DEBF-BCF6-48E7-847C-72C31A1DBCBD}" />

</aims:putMetadataRelationship>
```

moveLayer

Used in: mapping
Parent elements: aims:map

```
<aims:moveLayer
    Attributes that set values:
    fromLayerId ="string"
    toLayerId ="string"
>

    No Child Elements
</aims:moveLayer >
```

Bold: Attribute or child element is required.

Description:

Moves specified layer in Layers collection to another location in the collection.

Restrictions:

None

Notes:

None

Attribute descriptions for moveLayer:

Attribute	Usage
fromLayerId	Current location of the object to move.
toLayerId	New location to place the object.

Examples for moveLayer:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>"
name="<%=myService%>" />

<aims:map id="myMap" legend="myLegend" envelope="myEnvelope"
serviceId="<%=myMapService%>" width="300" height="100" background="100,140,230"
transcolor="0,0,0" bufferImage="true" bufferRegion="true">

<aims:moveLayer fromLayerId="0" toLayerId="1" />

</aims:map>
```

moveLayerToBottom

Used in: mapping
Parent elements: aims:map

```
<aims:moveLayerToBottom
    Attributes that set values:
    layerId ="string"
>
    No Child Elements
</aims:moveLayerToBottom >
```

Bold: Attribute or child element is required.

Description:

Moves specified layer in Layers collection to the end of the collection.

Restrictions:

None

Notes:

None

Attribute descriptions for moveLayerToBottom:

Attribute	Usage
layerId	The layer in the collection to move to the bottom.

Examples for moveLayerToBottom:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>"
name="<%=myService%>" />

<aims:map id="myMap" legend="myLegend" envelope="myEnvelope"
serviceId="<%=myMapService%>" width="300" height="100" background="100,140,230"
transcolor="0,0,0" bufferImage="true" bufferRegion="true">

<aims:moveLayerToBottom layerId="0" />

</aims:map>
```

moveLayerToTop

Used in: mapping
Parent elements: aims:map

```
<aims:moveLayerToTop
    Attributes that set values:
    layerId ="string"
>

    No Child Elements
</aims:moveLayerToTop >
```

Bold: Attribute or child element is required.

Description:

Moves specified layer in Layers collection to the beginning of the collection.

Restrictions:

None

Notes:

None

Attribute descriptions for moveLayerToTop:

Attribute	Usage
layerId	The layer in the collection to move to the top.

Examples for moveLayerToTop:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>"
name="<%=myService%>" />

<aims:map id="myMap" legend="myLegend" envelope="myEnvelope"
serviceId="<%=myMapService%>" width="300" height="100" background="100,140,230"
transcolor="0,0,0" bufferImage="true" bufferRegion="true">

<aims:moveLayerToTop layerId="3" />

</aims:map>
```

nestedSearch

Used in: metadata
Parent elements: aims:search

```
<aims:nestedSearch
    Attributes that set values:
    operator ="and | or"
>

    <documentInfo... />
    <searchEnvelope... />
    <subset... />
    <textQuery... />
    <updated... />
    <valueQuery... />

</aims:nestedSearch >
```

Bold: Attribute or child element is required.

Description:

Temporarily changes the nested search criteria for a metadata document from an "and" to an "or".

Restrictions:

Must be nested within aims:search.

Notes:

None

Attribute descriptions for nestedSearch:

Attribute	Usage
operator	Value to use in the search criteria of a nested search.

Examples for nestedSearch:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300"/>

<aims:search  id="searchResults" connectionId="<%=myConnection%>" service="serviceName"
sort="name" operator="or">

  <aims:subset docId="{9DE86F30-F624-439F-B8C4-71CA3337DB59}" />
  <aims:documentInfo name="Parent" />

  <aims:nestedSearch operator="or">

    <aims:updated after="2002-01-24 11:55:47" before="2002-01-24 11:55:47" />

  </aims:nestedSearch>

</aims:search>
```

northArrow

Used in: mapping
Parent elements: aims:acetateObject

<aims:northArrow

Attributes that set values:

When using ArcMap Server:
x ="*double*"
y ="*double*"
angle ="0.0 - 360.0" **[0]**
size ="*integer*" **[30]**
type ="1 - 8" **[1]**

When using Image Server:
x ="*double*"
y ="*double*"
angle ="0.0 - 360.0" **[0]**
antialiasing ="true | false" **[false]**
outline ="0,0,0 - 255,255,255"
overlap ="true | false" **[true]**
shadow ="0,0,0 - 255,255,255"
size ="*integer*" **[30]**
transparency ="0.0 - 1.0" **[1.0]**
type ="1 - 8" **[1]**
>

No Child Elements
</aims:northArrow >

Bold: Attribute or child element is required.

Description:

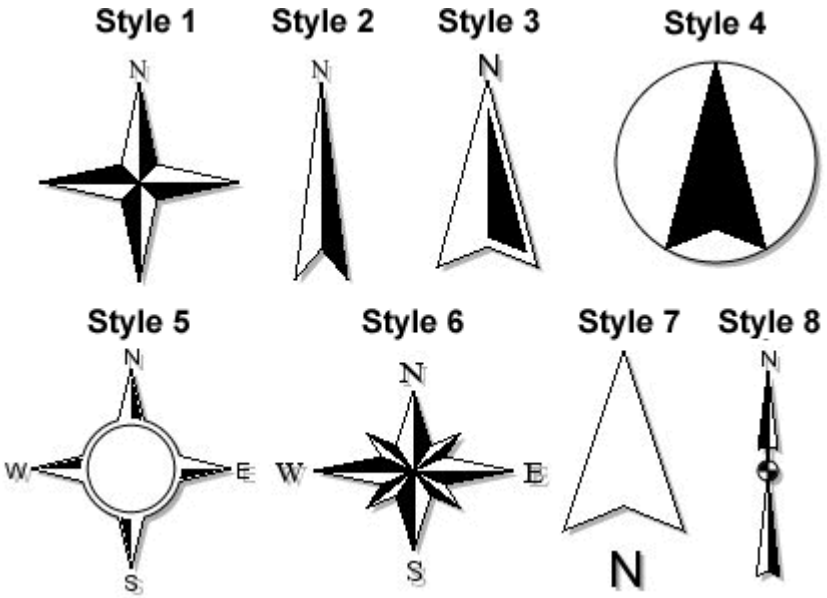
Places a North arrow on the acetate layer of the map.

Restrictions:

None

Notes:

- The following North arrow types are available:



- Aims:northarrow does not support custom arrows. Use aims:rasterMarkerSymbol or aims:truetypeMarkerSymbol instead.

Attribute descriptions for northArrow:

Attribute	Usage
angle	Angle of the North arrow in degrees. 0 degrees points to the north, and values increase moving clockwise.
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
outline	Outline color using RGB values.
overlap	Determines if labels can overlap this symbol. When "true", labels can overlap. When "false", labels will not overlap the symbol. If labels are not drawing as expected, check if *overlap* is set to "false" for this symbol or any other symbol in the service.
shadow	Shadow color using RGB values.
size	Arrow size in pixels.
transparency	Value to set percentage of transparency. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.
type	Value representing North arrow type.
x	X-coordinate for North arrow placement location.
y	Y-coordinate for North arrow placement location.

Examples for northArrow:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="SantaClara"
/>

<aims:map id="myMap" serviceId="<%=myMapService%>">
  <aims:createLayer layerId="northLayer" name="AcetateLayer" type="acetate"
visible="true">
    <aims:acetateObject units="pixel">
      <aims:northArrow x="50" y="50" type="3" shadow="230,230,230" />
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>
<img src="<%=(myMap)%>" />
```

other

Used in: mapping
Parent elements: aims:valueMapLabelRenderer aims:valueMapRenderer

```
<aims:other
    Attributes that set values:
    label ="string"
>

    When parent element is aims:valueMapLabelRenderer:
    <calloutMarkerSymbol... />  [Or]
    <chartSymbol... />  [Or]
    <rasterShieldSymbol... />  [Or]
    <shieldSymbol... />  [Or]
    <textSymbol... />  [Or]

    When parent element is aims:valueMapRenderer:
    <gradientFillSymbol... />  [Or]
    <hashLineSymbol... />  [Or]
    <rasterFillSymbol... />  [Or]
    <rasterMarkerSymbol... />  [Or]
    <simpleLineSymbol... />  [Or]
    <simpleMarkerSymbol... />  [Or]
    <simplePolygonSymbol... />  [Or]
    <trueTypeMarkerSymbol... />  [Or]

</aims:other >
```

Description:

Used with value maps as the default for rendering symbols that do not meet the criteria for any values in aims:range or aims:exact.

Restrictions:

Not valid with ArcMap Server.

Notes:

None

Attribute descriptions for other:

Attribute	Usage
label	Label for legend.

Examples for other:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />
<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="350">
<aims:layer layerId="0" visible="true" >
    <aims:valueMapRenderer lookUpField="TRACT">
        <aims:exact label="Tract#5121" value="5121" >
            <aims:simplePolygonSymbol fillColor="255,0,0" fillTransparency="1.0"/>
        </aims:exact>
        <aims:exact label="Tract#511898" value="511898" >
            <aims:simplePolygonSymbol fillColor="0,255,0" fillTransparency="1.0"/>
        </aims:exact>
        <aims:other label="other">
            <aims:simplePolygonSymbol fillColor="0,0,255" fillTransparency="1.0"/>
        </aims:other>
    </aims:valueMapRenderer>
</aims:layer>
</aims:map>
<img src="<%=(myMap)%>" />
```

pan

Used in: mapping
Parent elements: aims:map

```
<aims:pan
    Attributes that set values:
    direction ="1 - 8"
    step ="0.0 - NNN"
>

    No Child Elements
</aims:pan >
```

Bold: Attribute or child element is required.

Description:

Pans to a section of the map by a specified step and direction.

Restrictions:

None

Notes:

None

Attribute descriptions for pan:

Attribute	Usage
direction	7 West 8 Northwest
step	The amount of directional shift.

Examples for pan:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />

<aims:mapService id="myService" connectionId="<%=myTCPConnection%>" name="serviceName"/>

<aims:map id="myMapURL" serviceId="<%=myService%>">
  <aims:pan step="5" direction="8" />
</aims:map>

<img src="<%=(myMapURL)%>" />
```

point

Used in: mapping
Parent elements: aims:acetateObject aims:displayFeatures aims:filter

```
<aims:point
    Attributes that set values:
    coords ="double"
>


    When parent element is aims:acetateObject:
    <rasterMarkerSymbol... />  [Or]
    <simpleMarkerSymbol... />  [Or]
    <truetypeMarkerSymbol... />  [Or]

    When parent element is aims:filter or aims:displayFeatures:
        No Child Elements

</aims:point >
```

Bold: Attribute or child element is required.

Description:

Draws a point as an acetate layer on the map or is used as a filter to select features.

Restrictions:

Symbols, as nested tags inside point, can only be used when point is part of an acetate layer. They cannot be used when point is used in a filter.

Notes:

None

Attribute descriptions for point:

Attribute	Usage
coords	X,y coordinates representing one or more points. Coordinate x,y values are separated by white space, and coordinate pairs are separated by a semicolon by default. The separators can be changed by using aims:separators.

Examples for point:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />
<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="serviceName"
/>
<aims:map id="myMap" serviceId="<%=myMapService%>">
  <aims:createLayer layerId="pointLayer" type="acetate" name="point" visible="true" >
    <aims:acetateObject units="pixel">
      <aims:point coords="100 150;150 150;200 150">
        <aims:simpleMarkerSymbol width="25" type="star" color="255,255,255"
outline="0,0,0" shadow="0,0,0" />
      </aims:point>
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>
<img src="<%=(myMap)%>" />
```

polygon

Used in: mapping
Parent elements: aims:acetateObject aims:displayFeatures aims:filter

```
<aims:polygon
    Attributes that set values:
    coords ="x1 y1;...xn yn"
    name ="string"
    polygonId ="string"
>

    When parent element is aims:acetateObject:
    (m) <hole... />
    <gradientFillSymbol... />  [Or]
    <hashLineSymbol... />  [Or]
    <rasterFillSymbol... />  [Or]
    <rasterMarkerSymbol... />  [Or]
    <simpleLineSymbol... />  [Or]
    <simpleMarkerSymbol... />  [Or]
    <simplePolygonSymbol... />  [Or]
    <truetypeMarkerSymbol... />  [Or]

    When parent element is aims:filter or aims:displayFeatures:
    (m) <hole... />

</aims:polygon >
```

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Draws a polygon as an acetate layer on the map or is used as a filter to select features.

Restrictions:

- Symbols, as nested tags inside polygon, can only be used when polygon is part of an acetate layer. They cannot be used when polygon is used in a filter.
- The polygon must be closed.

Notes:

None

Attribute descriptions for polygon:

Attribute	Usage
coords	X,y coordinates representing a polygon. Coordinate x,y values are separated by white space, and coordinate pairs are separated by a semicolon by default. The separators can be changed by using aims:separators.
name	Name of the polygon.
polygonId	Identifier for the polygon.

Examples for polygon:

Example 1:

```
<aims:tcpConnection id="mytcpConnection" port="5300" host="myMachine" />
<aims:mapService id="myMapService" connectionId="<%=mytcpConnection%>"
name="serviceName"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:createLayer layerId="idPolygon" name="polygon" type="acetate" visible="true">
    <aims:acetateObject units="pixel">
      <aims:polygon coords="10 110;400 110;400 160;10 160;10 110">
        <aims:gradientFillSymbol transparency="1.0" type="vertical" startColor="0,255,0"
finishColor="0,0,255" overlap="true" />
      </aims:polygon>
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>
<img src="<%=(myMap)%>" />
```

publishDocument

Used in: metadata
Parent elements: None

```
<aims:publishDocument
    Attributes that set values:
    service ="string"

    Attributes that pass objects:
    connectionId ="object"
    document ="object"
>

    No Child Elements
</aims:publishDocument >
```

Bold: Attribute or child element is required.

Description:

Publishes a metadata document from the metadata repository.

Restrictions:

This element can be used only by the owner established in aims:createDocument. Any other user will get an error message.

Notes:

None

Attribute descriptions for publishDocument:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
document	The Document object to publish.
service	The name of the metadata service to publish the document to.

Examples for publishDocument:

Example 1: Publishes a metadata document from the metadata repository.

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true"
username="publish" password="publish" />

<aims:getUUID id="myUUID" connectionId="<%=(myConnection)%>" service="serviceName" />
<%
String docId = myUUID.elementAt(0).toString();
%>

<aims:createDocument id="myDocument" docId="<%=(docId)%>" name="Document" folder="true"
/>

<aims:publishDocument connectionId="<%=(myConnection)%>" document="<%=(myDocument)%>"
service="serviceName" />
```

putMetadataRelationship

Used in: metadata
Parent elements: None

```
<aims:putMetadataRelationship

    Attributes that set values:
    docId ="string"
    service ="string"

    Attributes that pass objects:
    connectionId ="object"
>

    (m) <metadataChild... />  [And/Or]
    (m) <metadataSibling... />  [And/Or]

</aims:putMetadataRelationship >
```

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Establishes a relationship between a source metadata document and one or more child or sibling metadata documents.

Restrictions:

At least one occurrence of aims:metadataChild or aims:metadataSibling is required. One or both tags can be used multiple times.

Notes:

A child document is a subdocument to the current metadata document. A sibling document is a related document. When selected as a sibling, the document is listed with the parent document under a link called "Related Documents".

Attribute descriptions for putMetadataRelationship:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
docId	String used to uniquely identify a document. The client used to publish the metadata is responsible for creating the document ID. This ID is automatically assigned when using ArcCatalog. If another client is used, aims:getUUID can be used to request a valid ID. The format for an ID is the following: {HHHHHHHH-HHHH-HHHH-HHHH-HHHHHHHHHHHH} where H is a hexadecimal digit (0–9,a–f,A–F). The ID is limited to 38 characters.
service	The name of the metadata service that contains the published document.

Examples for putMetadataRelationship:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" username="publish"
password="publish" />

<aims:putMetadataRelationship connectionId="<%=(myConnection)%>" service="serviceName"
docId="{E6F7632B-04CE-4B50-A995-887DD144F4DC}">

  <aims:metadataChild docId="{9F582D73-7001-4A4B-A806-0FB8392E1F8F}" />
  <aims:metadataSibling docId="{C7F0DEBF-BCF6-48E7-847C-72C31A1DBCBD}" />

</aims:putMetadataRelationship>
```

putMetadataSemantic

Used in: **metadata**
Parent elements: None

```
<aims:putMetadataSemantic
    Attributes that set values:
    service ="string"

    Attributes that pass objects:
    connectionId ="object"
>

    (m) <semanticPair... />

</aims:putMetadataSemantic >
```

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Framework for establishing a relationship between a tag and the numeric code defining semantics from the GEO Attribute Set.

Restrictions:

None

Notes:

None

Attribute descriptions for putMetadataSemantic:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
service	The name of the metadata service to establish the semantic definition.

Examples for putMetadataSemantic:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" username="publish"
password="publish" />

<aims:putMetadataSemantic connectionId="<%=(myConnection)%>" service="serviceName" >

  <aims:semanticPair tag="idinfo/citation/citeinfo/title" zcode="4" />

</aims:putMetadataSemantic>
```

range

Used in: mapping
Parent elements: aims:valueMapLabelRenderer aims:valueMapRenderer

<aims:range

Attributes that set values:
lower ="*string, numeric, or date*"
upper ="*string, numeric, or date*"
equality ="all | upper | lower | none" **[lower]**
label ="*string*"
>

*When parent element is **aims:valueMapLabelRenderer:***
<calloutMarkerSymbol... /> *[Or]*
<chartSymbol... /> *[Or]*
<rasterShieldSymbol... /> *[Or]*
<shieldSymbol... /> *[Or]*
<textSymbol... /> *[Or]*

*When parent element is **aims:valueMapRenderer:***
<gradientFillSymbol... /> *[Or]*
<hashLineSymbol... /> *[Or]*
<rasterFillSymbol... /> *[Or]*
<rasterMarkerSymbol... /> *[Or]*
<simpleLineSymbol... /> *[Or]*
<simpleMarkerSymbol... /> *[Or]*
<simplePolygonSymbol... /> *[Or]*
<trueTypeMarkerSymbol... /> *[Or]*

</aims:range >

Bold: Attribute or child element is required.

Description:

Used with value map for matching a range of values within a specified field in a database. When a match occurs, the symbol is drawn as specified for each range.

Restrictions:

Not valid with ArcMap Server.

Notes:

If there are leading or trailing blanks in a field value, they are trimmed before a comparison is made. For example, a field value of " Hello " is interpreted as "Hello".

Attribute descriptions for range:

Attribute	Usage
equality	Defines the upper and lower bounds of each range. If "all" is used, then lower <= value <= upper. If "upper" is used, lower < value <= upper. If "lower" is used, lower <= value < upper.
label	Label for legend.
lower	Lower value of range; can be a numeric, string, or date value.
upper	Upper value of range; can be a numeric, string, or date value.

Examples for range:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />
<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="350">
  <aims:layer layerId="6" visible="true" >
    <aims:valueMapLabelRenderer lookUpField="POP" labelField="NAME" >
        <aims:range lower="0" upper="261892" equality="none" label="less than 261892">
```

```
            <aims:textSymbol font="Arial" fontSize="12" fontColor="255,0,255" />
        </aims:range>
        <aims:range lower="261893" upper="522070" equality="none" label="261893 to
522070">
            <aims:textSymbol font="Arial" fontSize="16" fontColor="0,255,0" />
        </aims:range>
        <aims:range lower="522071" upper="782249" equality="none" label="522071 than
782249">
            <aims:textSymbol font="Arial" fontSize="20" fontColor="255,0,0" />
        </aims:range>
    </aims:valueMapLabelRenderer>
  </aims:layer>
</aims:map>
<img src="<%=(myMap)%>" />
```

rasterFillSymbol

Used in: mapping
Parent elements: aims:buffer aims:displayFeatures aims:exact aims:other aims:polygon aims:range aims:simpleRenderer aims:targetLayer

```
<aims:rasterFillSymbol
    Attributes that set values:

    When using ArcMap Server:
    image ="path to image file"
    url ="url string"

    When using Image Server:
    image ="path to image file"
    url ="url string"
    antialiasing ="true | false"  [false]
    overlap ="true | false"  [true]
    transparency ="0.0 - 1.0"  [1.0]
>

    No Child Elements
</aims:rasterFillSymbol >
```

Bold: Attribute or child element is required.

Description:

Fills polygon features with specified image.

Restrictions:

In ArcMap Image Services, symbol is valid only in acetate layers.

Notes:

Acceptable image formats are JPG and GIF.

Attribute descriptions for rasterFillSymbol:

Attribute	Usage
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
image	Full pathname to image. ArcIMS Spatial Server uses this pathname to find the image and add it to the map. UNC pathnames can be used (\\myComputer\arcims\output).
overlap	Determines if labels can overlap this symbol. When "true", labels can overlap. When "false", labels will not overlap the symbol. If labels are not drawing as expected, check if *overlap* is set to "false" for this symbol or any other symbol in the service.
transparency	Value to set percentage of transparency. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.
url	URL used by client to retrieve image.

Examples for rasterFillSymbol:

Example 1:

```
<aims:tcpConnection id="myTCPConnection"  host="myMachine" port="5300" debug="true"/>
<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="serviceName"
/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>">
<aims:layer layerId="0" visible="true">
  <aims:displayFeatures whereExpression="CNTRY_NAME = 'India'"  zoomToFeatures="true">
    <aims:rasterFillSymbol transparency="0.5" overlap="true"
url="http://myMachine/output/Prodimg.jpg" image="D:/ESRI/Output/Prodimg.jpg"
antialiasing="false" />
  </aims:displayFeatures>
</aims:layer>
</aims:map>
<img src="<%=(myMap)%>" />
```

rasterMarkerSymbol

Used in: mapping
Parent elements: aims:buffer aims:displayFeatures aims:exact aims:line aims:other aims:point aims:polygon aims:range aims:simpleRenderer aims:targetLayer

<aims:rasterMarkerSymbol

Attributes that set values:

When using ArcMap Server:
image ="path to image file"
url ="url string"
size ="1,1 - N,N"

When using Image Server:
image ="path to image file"
url ="url string"
antialiasing ="true | false" **[false]**
hotSpotX ="*integer*" **[centered]**
hotSpotY ="*integer*" **[centered]**
overlap ="true | false" **[true]**
shadow ="0,0,0 - 255,255,255"
sizex ="*integer*"
sizey ="*integer*"
transparency ="0.0 - 1.0" **[1.0]**
useCentroid ="true | false" **[false]**
>

No Child Elements
</aims:rasterMarkerSymbol >

Bold: Attribute or child element is required.

Description:

Symbolizes point features using the specified raster image.

Restrictions:

In ArcMap Image Services, symbol is valid only in acetate layers.

Notes:

Acceptable image formats are JPG and GIF.

Attribute descriptions for rasterMarkerSymbol:

Attribute	Usage
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
hotSpotX	Determines where marker symbol is placed in relation to actual x and y location of the point the marker symbol represents. A hotspot of 0,0 places the point at the top left corner of the marker symbol. X-coordinates are positive and measured in pixels. The default hotspot centers the marker symbol over the point based on its actual size. For example, if a marker symbol is 16x16 pixels, the default location is 8,8. If the size attribute is set to 32x32, the default hotspot center is still 8,8.
hotSpotY	Determines where marker symbol is placed in relation to actual x and y location of the point the marker symbol represents. A hotspot of 0,0 places the point at the top left corner of the marker symbol. Y-coordinates are positive and measured in pixels. The default hotspot centers the marker symbol over the point based on its actual size. For example, if a marker symbol is 16x16 pixels, the default location is 8,8. If the size attribute is set to 32x32, the default hotspot center is still 8,8.
image	Full pathname to image. ArcIMS Spatial Server uses this pathname to find the image and add it to the map. UNC pathnames can be used (\\myComputer\arcims\output).
overlap	Determines if labels can overlap this symbol. When "true", labels can overlap. When "false", labels will not overlap the symbol. If labels are not drawing as expected, check if *overlap* is set to "false" for this symbol or any other symbol in the service.
shadow	Shadow color using RGB values.

Attribute	Usage
sizex	Resizes marker symbol to new size in pixels. The default size is the actual width and height of the marker symbol. If *size="0,0"* is specified, the ImageServer writes a warning message to the log file and uses the default size settings.
sizey	Resizes marker symbol to new size in pixels. The default size is the actual width and height of the marker symbol. If *size="0,0"* is specified, the ImageServer writes a warning message to the log file and uses the default size settings.
transparency	Value to set percentage of transparency. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.
url	
useCentroid	By default, a marker symbol used on polygon layers draws markers at all polygon vertices. If *useCentroid* is "true", marker is placed in the centroid of the polygon. If multiple polygon parts exist, the marker falls on the part with the biggest area.

Examples for rasterMarkerSymbol:

Example 1:

```
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:separators  ts="," cs=" " />
  <aims:createLayer id="myLayerId" name="point" type="acetate" visible="true">
    <aims:acetateObject>
      <aims:point coords="-122.495 37.788,-122.484 37.788">
        <aims:rasterMarkerSymbol url="http://mymachine.domain.com/website/color.gif"
image="C:/ArcIMS/WebSite/color.gif" />
      </aims:point>
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>
```

rasterShieldSymbol

Used in: mapping
Parent elements: aims:exact aims:other aims:range aims:simpleLabelRenderer

```
<aims:rasterShieldSymbol
    Attributes that set values:
    image ="path to image file"
    url ="url string"
    antialiasing ="true | false"  [false]
    font ="Any system font"  [Arial]
    fontColor ="0,0,0 - 255,255,255"  [0,0,0]
    fontSize ="1 - NNN"  [12]
    fontStyle ="regular | bold | italic | underline | outline | bolditalic"  [regular]
    labelMode ="full | numericonly"  [numericonly]
    printMode ="titlecaps | allupper | alllower | none"  [none]
    shadow ="0,0,0 - 255,255,255"
    textPosition ="0,0 - N,N"
    transparency ="0.0 - 1.0"  [1.0]
>

    No Child Elements
</aims:rasterShieldSymbol >
```

Bold: Attribute or child element is required.

Description:

A raster shield is a user-specified image and is used as a custom shield to identify roads or other line features. The text associated with the image comes from a specified field and is placed on top of the image.

Restrictions:

- Works only with line features.
- Not valid with ArcMap Server.

Notes:

- Acceptable image formats are JPG and GIF.
- The field for text is specified in the associated label renderer element.
- The image needs to be wide enough to support any long text. The image will not automatically resize to accommodate long strings.

Attribute descriptions for rasterShieldSymbol:

Attribute	Usage
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
font	Font name. The name is case sensitive. If font name uses "&", use "&" instead. For example, ESRI Transportation & Civic should be written as ESRI Transportation & Civic.
fontColor	Font color using RGB values.
fontSize	Font size.
fontStyle	Font style.
image	Full pathname to image. ArcIMS Spatial Server uses this pathname to find the image and add it to the map. UNC pathnames can be used (\\myComputer\arcims\output).
labelMode	Determines what value is drawn on raster shield. If "full" is used, the entire field value, such as I-80, is displayed. If "numericonly" is used, only numbers within the field are displayed. For example, I-80 is displayed as 80.
printMode	Determines how labels are printed. If "none" is used, no change is made to the label: Welcome to ArcIMS. If "alllower" is used, all letters are lowercase: welcome to arcims. If "allupper" is used, all letters are uppercase: WELCOME TO ARCIMS. If "titlecaps" is used, the first letter of each word in a label is uppercase and everything else is lowercase: Welcome To Arcims.
shadow	Shadow color using RGB values.
textPosition	Determines where text is placed in relation to shield image. The coordinate 0,0 is in the bottom left corner, and x and y are positive and measured in pixels. If attribute is not used, then text is placed in center of shield image.
transparency	Value to set percentage of transparency. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.
url	URL used by client to retrieve image.

Examples for rasterShieldSymbol:

Example 1:

```
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:layer layerId="0" visible="true">
    <aims:simpleLabelRenderer field="AREA">
      <aims:rasterShieldSymbol transparency="1.0" font="Arial" fontStyle="bolditalic"
fontSize="16" fontColor="255,255,255" shadow="125,125,125"
url="http://mymachine.domain.com/website/shield.gif" image="C:/ArcIMS/WebSite/shield.gif"
/>
    </aims:simpleLabelRenderer>
  </aims:layer>
</aims:map>
```

recordset

Used in: mapping
Parent elements: aims:layer

```
<recordset >
    No Attributes

    <iterateEnvelope... />
    <iterateGeometry... />
    <iterateRecordset... />
    <iterateTableDesc... />

</aims:recordset >
```

Description:

Main tag that holds information about records of a layer.

Restrictions:

None

Notes:

None

Examples for recordset:

Example 1:

```
<aims:tcpConnection id="myTCPConnection"  host="myMachine" port="5300" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>"
name="<%=myService%>" loadRecordset="true" loadEnvelope="true" loadExtensions="true"
loadRenderer="true" />

<aims:map id="myMap" legend="myLegend" envelope="myEnvelope"
serviceId="<%=myMapService%>" width="500" height="300">
  <aims:separators  ts="," cs=" " />
  <aims:layer layerId="0" visible="true">
    <aims:filter envelope="true" geometry="true" globalEnvelope="true">
      <aims:line coords="-122.483 37.769,-122.482 37.763" />
    </aims:filter>
    <aims:recordset>
      <aims:iterateTableDesc fieldName="fName" fieldType="fType" fieldPrecision="fPrec"
fieldLength="fLen">
        <%=(fName)%>
        <%=(fType)%>
        <%=(fPrec)%>
        <%=(fLen)%> <br>
      </aims:iterateTableDesc>
    </aims:recordset>
  </aims:layer>
</aims:map>
```

removeLayer

Used in: mapping
Parent elements: aims:map

```
<aims:removeLayer
    Attributes that set values:
    layerId ="string"
>
    No Child Elements
</aims:removeLayer >
```

Bold: Attribute or child element is required.

Description:

Removes a layer from the Layers collection.

Restrictions:

None

Notes:

None

Attribute descriptions for removeLayer:

Attribute	Usage
layerId	The layer in the layer collection to remove.

Examples for removeLayer:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />

<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>"
name="<%=myService%>" />

<aims:map id="myMap" legend="myLegend" envelope="myEnvelope"
serviceId="<%=myMapService%>" width="300" height="100" background="100,140,230"
transcolor="0,0,0" bufferImage="true" bufferRegion="true">

  <aims:removeLayer layerId="1" />

</aims:map>
```

removeVirtualServer

Used in: administration
Parent elements: None

```
<aims:removeVirtualServer
    Attributes that pass objects:
    connectionId ="object"
    virtualServer ="object"

    Attributes that return values:
    error ="string"
>

    No Child Elements
</aims:removeVirtualServer >
```

Bold: Attribute or child element is required.

Description:

Removes a Virtual Server.

Restrictions:

None

Notes:

None

Attribute descriptions for removeVirtualServer:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
error	Returns an error string if any occurred. Returns "Success" if no error.
virtualServer	The VirtualServer object to remove. Use aims:getVirtualServer or aims:createVirtualServer to retrieve the Virtual Server information.

Examples for removeVirtualServer:

Example 1:

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://brugge"
      role="admin"
      debug="true"
      username="admin"
      password="admin"
/>

<aims:getVirtualServer id="myServer" connectionId="<%=myhttpConnection%>"
name="FeatureServer2" error="getError" />

<aims:removeVirtualServer virtualServer="<%=myServer%>"
connectionId="<%=myhttpConnection%>" error="removeError" />
```

renameMetadata

Used in: metadata
Parent elements: None

```
<aims:renameMetadata
    Attributes that set values:
    docId ="string"
    newName ="string"
    service ="string"

    Attributes that pass objects:
   connectionId ="object"
>

    No Child Elements
</aims: renameMetadata >
```

Bold: Attribute or child element is required.

Description:

Changes the name of a metadata document.

Restrictions:

None

Notes:

None

Attribute descriptions for renameMetadata:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
docId	String used to uniquely identify a document. The client used to publish the metadata is responsible for creating the document ID. This ID is automatically assigned when using ArcCatalog. If another client is used, aims:getUUID can be used to request a valid ID. The format for an ID is the following: {HHHHHHHH-HHHH-HHHH-HHHH-HHHHHHHHHHHH} where H is a hexadecimal digit (0–9,a–f,A–F). The ID is limited to 38 characters.
newName	New name that identifies the dataset.
service	The name of the metadata service containing the document to delete.

Examples for renameMetadata:

Example 1:

```
<%@taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>Publish - renameMetadata</title>
</head>
<body>
<aims:httpConnection id="myConnection" url="http://localhost" username="publish"
password="publish" />

<%-- Rename Metadata --%>

<aims:renameMetadata connectionId="<%=(myConnection)%>" docId="{D589B205-5AA8-4A32-A7A8-
532F06CE4B0C}" newName="States" service="Metadata" />

</body>
</html>
```

request

Used in: mapping metadata administration
Parent elements: None

```
<aims:request
    Attributes that set values:
    request ="string"
    service ="string"
    password ="string"  [null]
    username ="string"  [null]

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    id ="string"
>

    No Child Elements
</aims:request >
```

Bold: Attribute or child element is required.

Description:

Sends an ArcXML request to an established server connection.

Restrictions:

None

Notes:

None

Attribute descriptions for request:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
id	Returns a response string with the reference value.
password	Overrides password set in aims:httpConnection or tcpConnection.
request	ArcXML request string. Refer to the *ArcXML Programmer's Reference Guide* on how to construct requests.
service	The name of the ArcIMS service.
username	Overrides username set in aims:httpConnection or tcpConnection.

Examples for request:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />
<aims:request  id="myResponse" connectionId="<%=myTCPConnection%>"
service="world"  request="<%=requestString%>" />
```

scale

Used in: mapping
Parent elements: aims:dataframe

```
<aims:scale
    rf ="double"
    x ="double"
    y ="double"
>
    No Child Elements
</aims:scale >
```

Bold: Attribute or child element is required.

Description:

Defines the data frame scale and center point.

Restrictions:

None

Notes:

None

Attribute descriptions for scale:

Attribute	Usage
rf	Relative factor, such as 1:24,000. The value for *rf* in this case would be 24,000.
x	X-coordinate representing the center of the map.
y	Y-coordinate representing the center of the map.

Examples for scale:

Example 1:

```
<%@ taglib uri="arcims_taglib.tld" prefix="aims" %>
<html>
<head>
<title>JSP Connector Test</title>
<LINK STYLE="text/css" REL="stylesheet" HREF="font.css">
</head>
<body>
<aims:tcpConnection id="myConnection" host="localhost" port="5300" debug="true" />

<aims:arcMapService id="arcmyDataframeCollection" connectionId="<%=(myConnection)%>"
name="serviceName" error="arcmyError" />

<aims:getArcMapLayout dataframeCollection="<%=(arcmyDataframeCollection)%>" width="500"
height="700">
  <aims:envelope minx="0" miny="0" maxx="50" maxy="40" />
  <aims:layoutOutput type="jpg" />
  <aims:dataframe dataframeName="South East Asia">
    <aims:featureCoordSys id="102030" />
    <aims:filterCoordSys id="102030" />
    <aims:scale rf="3251279" x="-5079159.83" y="2146309.62"  />
  </aims:dataframe>
</aims:getArcMapLayout>

<aims:getArcMapLayoutAttribute id="arclayoutURL" value="LAYOUTURL"
dataframeCollection="<%=(arcmyDataframeCollection)%>" />
<img src="<%=(arclayoutURL)%>" /><br>

</body>
</html>
```

scalebar

Used in: mapping
Parent elements: aims:acetateObject

<aims:scalebar

Attributes that set values:

When using ArcMap Server:
x ="*double*"
y ="*double*"
barColor ="0,0,0 - 255,255,255" **[255, 162, 115]**
barWidth ="*integer*" **[5]**
font ="Any system font" **[Arial]**
fontColor ="0,0,0 - 255,255,255" **[0,0,0]**
fontSize ="*integer*" **[10]**
fontStyle ="regular | bold | italic | underline | outline | bolditalic" **[regular]**
mapUnits ="degrees | meters | feet" **[degrees]**
scaleUnits ="miles | feet | meters | kilometers" **[miles]**
screenLength ="*integer*"

When using Image Server:
x ="*double*"
y ="*double*"
antialiasing ="true | false" **[false]**
barColor ="0,0,0 - 255,255,255" **[255, 162, 115]**
barTransparency ="0.0 - 1.0" **[1]**
barWidth ="*integer*" **[5]**
distance ="*double*"
font ="Any system font" **[Arial]**
fontColor ="0,0,0 - 255,255,255" **[0,0,0]**
fontSize ="*integer*" **[10]**
fontStyle ="regular | bold | italic | underline | outline | bolditalic" **[regular]**
mapUnits ="degrees | meters | feet" **[degrees]**
mode ="cartesian | geodesic" **[cartesian]**

```
    outline ="0,0,0 - 255,255,255"  [255,255,255]
    overlap ="true | false"  [true]
    precision ="integer"  [0]
    round ="double"
    scaleUnits ="miles | feet | meters | kilometers"  [miles]
    screenLength ="integer"
    textTransparency ="0.0 - 1.0"  [1]
>

    No Child Elements
</aims:scalebar >
```

Bold: Attribute or child element is required.

Description:

Defines the look and feel of the scalebar in the acetate layer.

Restrictions:

None

Notes:

None

Attribute descriptions for scalebar:

Attribute	Usage
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
barColor	Scalebar color using RGB values.
barTransparency	Value to set percentage of transparency. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.
barWidth	Scalebar width in pixels.

Attribute	Usage
distance	Sets the length of the scalebar to always be the distance specified. The distance units are the same as the scale units.
font	Font name. The name is case sensitive. If font name uses "&", use "&" instead. For example, ESRI Transportation & Civic should be written as ESRI Transportation & Civic.
fontColor	Font color using RGB values.
fontSize	Font size.
fontStyle	Font style.
mapUnits	Data units on map.
mode	Used when the map units are in decimal degrees. When the mode is "geodesic", the Image Server takes into account the position on the globe when calculating the size of the scalebar symbol. When the mode is "cartesian", the Image Server uses the same calculation for the scale bar for all points on the globe. The calculation is made at the equator.
outline	Outline color using RGB values.
overlap	Determines if labels can overlap this symbol. When "true", labels can overlap. When "false", labels will not overlap the symbol. If labels are not drawing as expected, check if *overlap* is set to "false" for this symbol or any other symbol in the service.
precision	Number of decimal places.
round	Number of digits to round.
scaleUnits	Screen units.
screenLength	Scalebar length in pixels.
textTransparency	Value to set percentage of transparency. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.
x	X-coordinate for scalebar placement location.
y	Y-coordinate for scalebar placement location.

Examples for scalebar:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300"/>
<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="serviceName"
/>
<aims:map id="myMap" legend="myLegend" envelope="myEnvelope"
serviceId="<%=myMapService%>">
  <aims:createLayer layerId="scaleLayer" type="acetate" name="scalebar" visible="true" >
    <aims:acetateObject units="pixel">
      <aims:scalebar x="150" y="25" fontSize="5" barWidth="10" mapUnits="decimal_degrees"
scaleUnits="miles" distance="100.0" />
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>
<img src="<%=(myMap)%>"/>
```

scaleDependentRenderer

Used in: mapping
Parent elements: aims:createLayer aims:groupRenderer aims:layer aims:scaleDependentRenderer

```
<aims:scaleDependentRenderer
    Attributes that set values:
    lower ="string or double"
    upper ="string or double"
>

    <groupRenderer... />  [Or]
    <scaleDependentRenderer... />  [Or]
    <simpleLabelRenderer... />  [Or]
    <simpleRenderer... />  [Or]
    <valueMapLabelRenderer... />  [Or]
    <valueMapRenderer... />  [Or]

</aims:scaleDependentRenderer >
```

Bold: Attribute or child element is required.

Description:

Displays specified rendering information at certain scales. A layer can have different renderings depending on the current scale. For example, when zoomed out, you can draw a street layer one pixel in width. As you zoom farther in, you can draw the street layer eight pixels in width and in a different color.

Restrictions:

- Only one nested tag can be used inside aims:scaleDependentRenderer.
- Not valid with ArcMap Server.

Notes:

Scales are often referenced using a relative scale, such as 1:24,000. In this example, 1 meter equals 24,000 meters, or 1 inch equals 24,000 inches. Another way to reference a scale in ArcIMS is by calculating the number of map units per pixel. For this release, the scales used in aims:scaleDependentRenderer must be in map units per pixel.

Map units per pixel refers to the number of meters, feet, or decimal degrees represented by one pixel in a map. To convert from a relative scale to map units per pixel, the size of a pixel must first be calculated. The formula for finding the number of meters in a pixel is 0.0254 / dpi. The value 0.0254 is the number of meters in an inch, and dpi is the dpi set in the MapService or request. If no dpi is set in the MapService or request, the dpi is assumed to be 96. As an example of pixel size, if the dpi is 96, the pixel size is 0.0254 / 96 or 0.000265 m. To convert from a relative scale to map units per pixel follow the directions below:

1. If the scale is in **meters**. To calculate the number of meters per pixel, take the relative scale and multiply by 0.000265. For example, if the relative scale is 1:24,000, then the number of meters per pixel is 24,000 * 0.000265, or 6.36 meters.
2. If the scale is in **feet**. Do the calculation for meters (#1). Multiply the result by 3.28 (the number of feet in a meter). For example, if the number of meters per pixel is 6.36, the number of feet is 6.36 * 3.28, or 20.86 feet.
3. If the scale is in **decimal degrees**. For these calculations, the earth is assumed to be an exact circle with a circumference of 40030.174 km. One degree is 111.195 km (40030.174/360 degrees), or 111195 meters. To calculate the number of degrees, first do the calculation for meters (#1). Next, divide the result by 111195. For example, if the number of meters per pixel is 6.36, the number of degrees is 6.36 / 111195, or 0.0000571968.

Attribute descriptions for scaleDependentRenderer:

Attribute	Usage
lower	Minimum scale to display renderer using a relative scale, such as 1:24,000. Scale can also be calculated as the number of map units per pixel.
upper	Maximum scale to display renderer using a relative scale, such as 1:24,000. Scale can also be calculated as the number of map units per pixel.

Examples for scaleDependentRenderer:

Example 1: When using a relative scale.

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />
<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:layer layerId="5" visible="true">
    <aims:scaleDependentRenderer lower="1:1" upper="1:250000">
      <aims:simpleLabelRenderer field="NAME">
        <aims:textSymbol />
      </aims:simpleLabelRenderer>
    </aims:scaleDependentRenderer>
  </aims:layer>
</aims:map>
<img src="<%=(myMap)%>" />
```

Example 2: When using map units per pixel.

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />
<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:layer layerId="5" visible="true">
    <aims:scaleDependentRenderer upper="0.0000571968">
      <aims:simpleLabelRenderer field="NAME">
        <aims:textSymbol />
      </aims:simpleLabelRenderer>
    </aims:scaleDependentRenderer>
  </aims:layer>
</aims:map>
<img src="<%=(myMap)%>" />
```

sdeWorkspace

Used in: mapping
Parent elements: aims:createLayer

```
<aims:sdeWorkspace
    Attributes that set values:
    featureClass ="point | line | polygon | image"
    instance ="string"
    name ="string"
    password ="string"
    server ="string"
    user ="string"
    database ="string"
    encrypted ="true | false"  [false]
    geoIndexDir ="string"  [Windows: c:\temp and UNIX: /tmp]
>

    No Child Elements
</aims:sdeWorkspace >
```

Bold: Attribute or child element is required.

Description:

Defines an ArcSDE data source.

Restrictions:

- Must refer to an existing ArcSDE data source.
- Not valid with ArcMap Server.

Notes:

- It is recommended not to use aims:sdeWorkspace in a request. All workspaces should be included in the map configuration file.
- To add an ArcSDE workspace, <MAP dynamic="true" > must be set in the map configuration file.
- Aims:sdeWorkspace can be used to set up an ArcSDE direct connect. For more information on direct connect and installation instructions for use with ArcIMS, see the Direct_Connect folder in the documentation folder located on the ArcIMS CD–ROM.
- The following table summarizes whether the attributes *server* and *database* are required or optional when using ArcSDE or ArcSDE Direct Connect. It also shows an example of correct syntax for the attribute *instance*.

Database	Server	Instance	Database
Oracle with ArcSDE	Required	Port:5151	Optional
Oracle with Direct Connect	Optional	sde:oracle (Oracle8*i*) sde:oracle9i (Oracle9*i*)	Optional
SQL Server with ArcSDE	Required	Port:5151	Required
SQL Server with Direct Connect	Optional	sde:sqlserver	Required
Other Databases	Required	Port:5151	Optional

- The next table summarizes whether the attribute *user* is required, the correct syntax for *password*, and whether the password can be encrypted.

Database	User	Password	Can Encrypt Password
Oracle with ArcSDE	Required	MyPassword	Yes
Oracle with Direct Connect	Required	MyPassword @net8servicename	No
SQL Server with ArcSDE	Required	MyPassword	Yes
SQL Server with Direct Connect	Required	MyPassword	No
Other Databases	Required	MyPassword	Yes

- Passwords for ArcSDE datasets, by default, are not encrypted. In order to encrypt a password, you need to connect to the ArcSDE instance while in ArcIMS Author or ArcExplorer™ 4. Since you cannot connect to an ArcSDE Direct Connect layer using ArcIMS Author or ArcExplorer 4, the password cannot be encrypted for ArcSDE Direct Connect.
- If layers do not appear in a map, double-check that the workspace information is correct including the username and password.

Attribute descriptions for sdeWorkspace:

Attribute	Usage
database	ArcSDE database.
encrypted	When set to "true", the password for the ArcSDE instance is encrypted. If "false", the password is not encrypted.
featureClass	Layer type.
geoIndexDir	Directory where geocoding index is built. On Windows, the default directory is the "temp" directory. On UNIX, the default is /tmp.
instance	Port number for ArcSDE instance. For example, if the port for an ArcSDE dataset is 5151, then the *instance* is "port:5151". For ArcSDE direct connect, the correct value is "sde:<database_vendor>". For ArcIMS 4, the accepted values with Direct Connect are "sde:oracle" (Oracle8*i*), "sde:oracle9i" (Oracle9*i*), and "sde:sqlserver".
name	Workspace name; must be unique among all data sources.
password	Password to access the ArcSDE server. For Oracle ArcSDE Direct Connect, the password must be appended with "@<net8 service name>".
server	ArcSDE server.
user	Username to access the ArcSDE server.

Examples for sdeWorkspace:

Example 1:

```
<aims:tcpConnection id="mytcpConnection" port="5300" host="myMachine"/>
<aims:mapService id="myService" connectionId="<%=mytcpConnection%>" name="serviceName" />

<aims:map id="mapImage" serviceId="<%=(myService)%>"  width="500" height="350" >
  <aims:createLayer layerId="SDELayer" type="sde" name="states" visible="true" >
    <aims:simpleRenderer>
      <aims:simplePolygonSymbol fillColor="0,255,0" transparency="1.0" />
    </aims:simpleRenderer>
    <aims:sdeWorkspace name="SDEStates" instance="port:5151" server="entropy"
password="bolivia" user="bolivia" featureClass="polygon" />
    <aims:dataset name="BOLIVIA.STATES" type="polygon" workspace="SDEStates" />
  </aims:createLayer>
</aims:map>

<img src="<%=(mapImage)%>" />
```

search

Used in: metadata
Parent elements: None

```
<aims:search
    Attributes that set values:
    batchSize ="integer"
    folderMask ="1 - 7"  [7]
    fullOutput ="true | false"  [true]
    gndextent ="none | document | search"  [none]
    operator ="and | or"  [and]
    sort ="name | relevance | contenttype | local_area | global_area"  [name]
    sort2 ="name | relevance | contenttype | local_area | global_area"
    startBatchAt ="0 - NNN"  [0]
    startResult ="integer"  [0]

    Attributes that pass objects:
    connectionId ="object"
    service ="string"

    Attributes that return values:
    id ="string"
    numResults ="string"
    totalResults ="string"
>

    <documentInfo... />
    <nestedSearch... />
    <searchEnvelope... />
    <subset... />
    <textQuery... />
    <updated... />
    <valueQuery... />

</aims:search >
```

Bold: Attribute or child element is required.

Description:

Searches a specified list of metadata documents.

Restrictions:

None

Notes:

Use nested tags to build search criteria.

Attribute descriptions for search:

Attribute	Usage
batchSize	The maximum number of results to return.
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
folderMask	Specifies folder types to be returned in the response. The value of *foldermask* is an integer storing the sum of one or more of the following values: • 1 = Root document • 2 = "Normal" folder • 4 = Document For example, use "4" to return documents only. Use "7" (1 + 2 + 4) to return all types of folders. The different values are: • 1 = Root document only (1) • 2 = "Normal" folders only (2) • 3 = Root document and "normal" folders (1 + 2) • 4 = Documents only (4) • 5 = Root document and documents (1 + 4) • 6 = "Normal" folders and documents (2 + 4) • 7 = Root document, "normal" folders, and documents (1 + 2 + 4)

fullOutput	By default this attribute is "true". When "true", an XML file is generated, and a thumbnail and gnd file is created if available. When "false", the XML file, thumbnail, and gnd files are not created.
gndextent	The extent written to the GND file. When "none" is selected, the default extent of the service is used. For "document" the extent is taken from the metadata document. For "search" the extent is the search extent specified in the client, such as Metadata Explorer.
id	Reference name of DataCollection object.
numResults	Returns a string containing the number of results returned in this batch.
operator	Used to define the operator for the query.
service	The name of the metadata service.
sort	Preference for ordering results. "Name" orders the results alphabetically. "Relevance" lists results from highest to lowest relevance. "Contenttype" sorts and groups results by content type. "Localarea" lists results by area in ascending order. "Globalarea" lists results by area in descending order.
sort2	Sorts search results that were batched using startresult and maxresults. "Name" orders the results alphabetically. "Relevance" lists results from highest to lowest relevance. "Contenttype" sorts and groups results by content type. "Localarea" lists results by area in ascending order. "Globalarea" lists results by area in descending order.
startBatchAt	By default, all records meeting the search criteria are returned starting with record 0. This attribute allows a specified record as the start record.
startResult	Returns a string containing the starting index of the batched results.
totalResults	Returns a string containing the total number of results found.

Examples for search:

Example 1: Searches a specified list of metadata documents.

```
<aims:getSettings id="hostName" value="meta_host_name" />
<aims:getSettings id="portNumber" value="meta_port_number" />
<aims:getSettings id="serviceName" value="meta_service_name" />
<aims:getSettings id="batchSize" value="result_batch_size" />

<aims:tcpConnection id="myConnection" host="<%= hostName %>" port="<%= portNumber %>"
debug="true" />
```

```
<%
String startBatchAt = request.getParameter("startBatchAt"); // where to start in the
collection of results
if (startBatchAt == null) startBatchAt = "0";
%>

<%-- search for datasets containing the word water and having a content type of 002 (ISO
- Downloadable Data) --%>
<aims:search id="results" connectionId="<%= myConnection %>" service="<%= serviceName %>"
sort="local_area" sort2="contenttype" folderMask="4" totalResults="totalResults"
startResult="startResult" numResults="numResults" batchSize="<%= batchSize %>"
startBatchAt="<%= startBatchAt %>">

   <aims:textQuery tag="_fulltext" word="water" />

   <aims:nestedSearch operator="or">
      <aims:valueQuery equalTo="002"
tag="metadata/distInfo/distributor/distorTran/onLineSrc/orDesc" operators="or" />
   </aims:nestedSearch>

</aims:search>

<%-- loop through the result set --%>
<aims:iterateMetadata iterate="<%= results %>" size="resultsSize">

<%-- Get the dataset's title --%>
<aims:getElement id="theTitle" loadedElements="<%= theElements %>"
value="metadata/dataIdInfo/idCitation/resTitle" />

<P>
Title: <%= theTitle %>

</aims:iterateMetadata>
```

searchEnvelope

Used in: metadata
Parent elements: aims:nestedSearch aims:search

```
<aims:searchEnvelope
    Attributes that set values:
    maxx ="double"
    maxy ="double"
    minx ="double"
    miny ="double"
    spatialOperator ="within | overlaps | overlaps2 | fuzzywithin | fuzzyequals"  [fuzzywithin]
>

    No Child Elements
</aims:searchEnvelope >
```

Bold: Attribute or child element is required.

Description:

Constructs an envelope with minimum and maximum x,y coordinates when including geographic criteria in searches.

Restrictions:

None

Notes:

None

Attribute descriptions for searchEnvelope:

Attribute	Usage
maxx	Maximum x-coordinate in map units.
maxy	Maximum y-coordinate in map units.
minx	Minimum x-coordinate in map units.
miny	Minimum y-coordinate in map units.
spatialOperator	The restrictions on the envelope. • within—finds metadata documents where the document's envelope falls entirely within the specified search box. The document's spatial envelope can touch the search box boundary. • overlaps—finds metadata documents in which part or all of the metadata document's envelope falls within the specified search box. Anything the search box touches is found. • overlaps2—same as "overlaps", except that the metadata document's envelope cannot completely contain the search box. • fuzzywithin—finds metadata documents in which the document's spatial extent is roughly within the specified search box. The specified search box is expanded by 10 percent and returns metadata documents that fall within the expanded outer border and intersect the original search box. Edge touching is allowed. • fuzzyequals—finds metadata documents in which the document's spatial extent is roughly equal to the specified search box. The specified search box is expanded 10 percent inside and outside. Returns documents that are completely contained by the outer expanded search area and completely contain the inner expanded search area.

Examples for searchEnvelope:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300"/>

<aims:search  id="searchResults" connectionId="<%=myConnection%>" service="serviceName"
sort="name" operator="or">

  <aims:subset docId="{9DE86F30-F624-439F-B8C4-71CA3337DB59}" />
  <aims:documentInfo name="Parent" />

  <aims:nestedSearch operator="or">

    <aims:updated after="2002-01-24 11:55:47" before="2002-01-24 11:55:47" />
    <aims:searchEnvelope minx="-122.496415054188" miny="16.6302970237522" maxx="-
88.2643590144781" maxy="37.7949559424933" />

  </aims:nestedSearch>

</aims:search>
```

semanticPair

Used in: metadata
Parent elements: aims:putMetadataSemantic

```
<aims:semanticPair
    Attributes that set values:
    tag ="string"
    zcode ="string"
    attribute ="string"
>

    No Child Elements
</aims:semanticPair >
```

Bold: Attribute or child element is required.

Description:

Establishes a relationship between a tag and the numeric code defining semantics from the GEO Attribute Set.

Restrictions:

None

Notes:

None

Attribute descriptions for semanticPair:

Attribute	Usage
attribute	Name of an attribute for the element.
tag	Defines the location of an element inside an XML tree. For example, the element "title" might be inside an XML structure such as <idinfo> <citation> <citeinfo> **<title>...</title>** </citeinfo> </citation> </idinfo> To define "title", the string value for this attribute would be "idinfo/citation/citeinfo/**title**".
zcode	Numeric code defining semantics from the GEO Attribute Set.

Examples for semanticPair:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" username="publish"
password="publish" />

<aims:putMetadataSemantic connectionId="<%=(myConnection)%>" service="serviceName" >

  <aims:semanticPair tag="idinfo/citation/citeinfo/title" zcode="4" />

</aims:putMetadataSemantic>
```

separators

Used in: mapping
Parent elements: aims:map

```
<aims:separators
    Attributes that set values:
    cs ="string"  [white space (" ")]
    ts ="string"  [semicolon (";")]
>

    No Child Elements
</aims:separators >
```

Description:

Identifies separators used between x,y coordinates and coordinate pairs and as separators for lists of strings.

Restrictions:

Separators are limited to one UNICODE character in length.

Notes:

- Coordinate and tuple separators are used to separate x,y coordinates and coordinate pairs, respectively, with the *coords* attribute in aims:point, aims:line, aims:polygon, and aims:hole.
- The tuple separator can be used to separate lists of strings in aims:exact.

Attribute descriptions for separators:

Attribute	Usage
cs	Coordinate separator is used to separate an x-coordinate from a y-coordinate.
ts	Tuple separator is used to separate coordinate pairs and string lists.

Examples for separators:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />
<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName"/>
<aims:map id="mapImage" serviceId="<%=myMapService%>" width="500" height="300" >
  <aims:separators ts=";" cs="," />
  <aims:createLayer layerId="polygonLayer" type="acetate" visible="true" >
    <aims:acetateObject>
      <aims:polygon coords="-122.038,37.366;-121.575,37.366;-121.575,37.218;-
122.038,37.218;-122.038,37.366">
        <aims:gradientFillSymbol />
        <aims:hole coords="-121.897,37.338;-121.812,37.338;-121.812,37.307;-
121.897,37.307;-121.897,37.338" />
      </aims:polygon>
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>
<img src="<%=(mapImage)%>" />
```

setInput

Used in: mapping
Parent elements: aims:addressMatchInputs

```
<aims:setInput
    Attributes that set values:
    inputId ="string"
    value ="string"
>

    No Child Elements
</aims:setInput >
```

Bold: Attribute or child element is required.

Description:

Sets a geocode input value that matches an aims:addressMatchInputs *id*.

Restrictions:

None

Notes:

If the input IDs are not known, first iterate through the inputs using aims:addressMatchInputs. Use aims:addressMatchInputs again to set the input ID values.

Attribute descriptions for setInput:

Attribute	Usage
inputId	Name of input ID retrieved from AddressMatchInputs object.
value	The value to set the input ID.

Examples for setInput:

Example 1:

```
<aims:tcpConnection id="mytcpConnection" port="5300" host="myMachine" debug="true"/>
<aims:mapService id="myService" connectionId="<%=mytcpConnection%>" name="geoServiceName"
loadExtensions="true" />
<aims:map id="mapImage" serviceId="<%=myService%>" reset="false" >
  <aims:layer layerId="3" >
    <aims:addressMatchInputs id="myInputs" minScore="0" maxCandidates="100" >
      <aims:setInput inputId="STREET" value="380 New York St" />
      <aims:setInput inputId="ZONE" value="" />
      <aims:setInput inputId="CROSSSTREET" value="" />
    </aims:addressMatchInputs>
    <aims:iterateAddressMatchInputs inputs="<%=myInputs%>" count="myCount"
minScore="myMinScore" maxCandidates="myMax" style="myStyle" >
      <aims:getInputAttribute id="myDes" attribute="DESCRIPTION" />
      <aims:getInputAttribute id="myId" attribute="ID" />
      <aims:getInputAttribute id="myLabel" attribute="LABEL" />
      <aims:getInputAttribute id="myType" attribute="TYPE" />
      <aims:getInputAttribute id="myWidth" attribute="WIDTH" />
        <b>Description </b>: <%=myDes%> <br>
        <b>Id          </b>: <%=myId%> <br>
        <b>Label       </b>: <%=myLabel%> <br>
        <b>Type        </b>: <%=myType%> <br>
        <b>Width       </b>: <%=myWidth%> <br>
    </aims:iterateAddressMatchInputs>
    <aims:iterateAddressMatchResults inputs="<%=myInputs%>" count="resultsCount" >
      <aims:getResultAttribute id="myValue" attribute="VALUE" />
      <aims:getResultAttribute id="myScore" attribute="SCORE" />
      <aims:getResultAttribute id="myPoint" attribute="POINT" />
        <b>Value </b>: <%=myValue%> <br>
        <b>Score </b>: <%=myScore%> <br>
        <b>Point </b>: <%=myPoint%> <br>
    </aims:iterateAddressMatchResults>
    <b>Result count </b>: <%=resultsCount%>
  </aims:layer>
</aims:map>
```

shapeWorkspace

Used in: mapping
Parent elements: aims:createLayer

```
<aims:shapeWorkspace
    Attributes that set values:
    directory ="string"
    featureClass ="point | line | polygon"
    name ="string"
    codepage ="string"
    geoIndexDir ="string"  [same as directory with shapefile]
    shared ="true | false"  [false]
>

    No Child Elements
</aims:shapeWorkspace >
```

Bold: Attribute or child element is required.

Description:

Defines a shapefile data source.

Restrictions:

- Must refer to an existing data source.
- Not valid with ArcMap Server.

Notes:

- It is recommended not to use aims:shapeWorkspace in a request. All workspaces should be included in the map configuration file.
- To add a shapefile workspace, <MAP dynamic="true" > must be set in the map configuration file.

Attribute descriptions for shapeWorkspace:

Attribute	Usage
codepage	Defines the codepage if it is not defined in the DBF header. The value is the name of the ICU transcoder. Examples include cp1252 and UTF8. The value used in codepage is valid only if the LDID byte in the DBF header is "0". If the LDID byte is not "0", then the value used in codepage is ignored, and the value used in the LDID byte is used instead.
directory	Directory containing shapefiles. UNC pathnames can be used (\\myComputer\shapefiledirectory).
featureClass	Layer type.
geoIndexDir	Directory where geocoding index is built.
name	Workspace name. Must be unique among all data sources.
shared	When set to "true", the ArcIMS Spatial Server checks if a shapefile has been modified outside ArcIMS. While a shapefile is being edited, ArcIMS sends a message to the ArcIMS Spatial Server log files saying that an update is in progress. When set to "false", the ArcIMS Spatial Server does not check whether a shapefile has been updated. Access to the shapefile is faster, but the integrity of the shapefile is at risk if it is modified. It is recommended to use "true" unless safeguards are in place to ensure that the shapefile is not edited. Note: In general, access to shapefiles is much faster if the shapefiles reside on the same machine as the ArcIMS Spatial Server. If the shapefiles are on a separate machine, access is faster when set to "false", but the integrity of the shapefile is at risk.

Examples for shapeWorkspace:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />
<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName"/>

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
<aims:createLayer layerId="topoq" type="shape" name="topoquad" visible="true" >
  <aims:simpleRenderer >
    <aims:simpleLineSymbol color="0,255,0" width="1" />
  </aims:simpleRenderer>
  <aims:shapeWorkspace name="workspace1"
```

```
directory="C:\\ArcIMS\\AXL\\TutorialData\\SantaClara" featureClass="line" />
  <aims:dataset name="sc_topoq24" type="polygon" workspace="workspace1" />
</aims:createLayer>
</aims:map>

<img src="<%=(myMap)%>" />
```

shieldSymbol

Used in: mapping
Parent elements: aims:exact aims:other aims:range aims:simpleLabelRenderer

```
<aims:shieldSymbol
    Attributes that set values:
    type ="interstate | usroad | rect | oval"
    antialiasing ="true | false"  [false]
    font ="Any system font"  [Arial]
    fontColor ="0,0,0 - 255,255,255"  [0,0,0]
    fontSize ="1 - NNN"  [12]
    fontStyle ="regular | bold | italic | underline | outline | bolditalic"  [regular]
    labelMode ="full | numericonly"  [numericonly]
    minSize ="1 - NNN"  [1]
    shadow ="0,0,0 - 255,255,255"
>

    No Child Elements
</aims:shieldSymbol >
```

Bold: Attribute or child element is required.

Description:

Symbol for drawing a predefined set of highway shields: U.S. Interstate, U.S. Highway, white rectangle, and white oval.

Restrictions:

- Works only with line features.
- Not valid with ArcMap Server.

Notes:

- The field for text is specified in the associated label renderer element.
- *LabelMode="full"* is designed for a maximum of four characters. If more than four characters are needed, aims:rasterShieldSymbol should be used.

Attribute descriptions for shieldSymbol:

Attribute	Usage
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
font	Font name. The name is case sensitive. If font name uses "&", use "&" instead. For example, ESRI Transportation & Civic should be written as ESRI Transportation & Civic.
fontColor	Font color using RGB values.
fontSize	Font size.
fontStyle	Font style.
labelMode	Determines what value is drawn on the shield. If "full" is used, the entire field value, such as I-80, is displayed. If "numericonly" is used, only numbers within the field are displayed. For example, I-80 is displayed as 80.
minSize	Sets shield size to minimum size in characters. By default, shield expands to length of text.
shadow	Shadow color using RGB values.
type	Symbol type.

Examples for shieldSymbol:

Example 1:

```
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:layer layerId="0" visible="true">
    <aims:simpleLabelRenderer field="AREA">
      <aims:shieldSymbol antialiasing="true" font="Arial" fontStyle="regular"
fontSize="10" type="interstate" />
    </aims:simpleLabelRenderer>
  </aims:layer>
</aims:map>
```

simpleLabelRenderer

Used in: mapping
Parent elements: aims:createLayer aims:groupRenderer aims:layer aims:scaleDependentRenderer

<aims:simpleLabelRenderer

Attributes that set values:
field ="*string*"
featureWeight ="no_weight | med_weight | high_weight" **[no_weight]**
howManyLabels ="one_label_per_name | one_label_per_shape | one_label_per_part"
labelBufferRatio ="*double*"
labelPriorities ="0,0,0,0,0,0,0,0 - 8,8,8,8,8,8,8,8 | LE_PlaceOnTopHorizontal" **[2,2,1,4,5,3,2,4]**
labelWeight ="no_weight | med_weight | high_weight" **[high_ weight]**
lineLabelPosition ="See table below for values" **[PlaceAbove]**
rotationalAngles ="*string*"
>

<calloutMarkerSymbol... /> *[Or]*
<chartSymbol... /> *[Or]*
<rasterShieldSymbol... /> *[Or]*
<shieldSymbol... /> *[Or]*
<textSymbol... /> *[Or]*

</aims:simpleLabelRenderer >
Bold: Attribute or child element is required.

Description:

Used for labeling features. A field is specified for labeling all features of a particular layer.

Restrictions:

- One symbol must be specified. Only one is allowed.
- Not valid with ArcMap Server.

Notes:

- When rotating symbols, the attribute *labelPriorities* always takes precedence over the attribute *rotationalAngles*. If you find that your labels are not rotating as expected, remove the *labelPriorities* attribute if it is present.
- If *subFields* is used in aims:filter for a layer, any fields used for labeling must be included in the subfields list.

Attribute descriptions for simpleLabelRenderer:

Attribute	Usage
featureWeight	Prioritizes the importance of features. The feature weight determines how important the feature labeled is for the label placement algorithm. If "no_weight" is specified, then the feature has no importance and can be labeled over. If "high_weight" is specified, then the feature has high importance and cannot be labeled over. Giving importance to features increases the complexity of the labeling problem and also the processing time.
field	Field containing text for labeling features. The field can be in the layer table or in a joined table. • For shapefiles with no joined tables, the field can be referenced using the short format. field="AREA" • For shapefiles with joined tables, the name of the joined table must be included along with the field. field="JOINEDTABLE.AREA" • For ArcSDE layers without joined tables, the field can be referenced using the short format. field="AREA" The full long name can also be used. field="ARCSDENAME.TABLE.AREA" • For ArcSDE layers with joined tables, joined fields must be referenced using the full long format. field="ARCSDENAME.TABLE.AREA" space: • Using a shapefile with no joined tables. The short format can be used for field names. field="CITY STATE_NAME"

	• Using a shapefile with joined tables. The name of the joined table must be included along with the field names. field="JOINEDTABLE.CITY JOINEDTABLE.STATE_NAME" • Using an ArcSDE layer without joined tables. The short format can be used. field="CITY STATE_NAME" The full long name can also be used. field="ARCSDENAME.TABLE.CITY ARCSDENAME.JOINEDTABLE.STATE_NAME" • Using an ArcSDE layer with joined tables. The full long format must be used. field="ARCSDENAME.TABLE.CITY ARCSDENAME.JOINEDTABLE.STATE_NAME"
howManyLabels	Determines how often a feature is labeled. • "One_label_per_part" labels all parts of a feature in the case of multipart features. For instance, the state of Hawaii consists of several islands, but they are grouped together as one polygon feature. Each island is labeled. • "One_label_per_name" labels once per feature name. When several features share the same name, only one label is shown. For example, if there is more than one polygon with the name "Residential", only one "Residential" polygon is labeled. • "One_label_per_shape" labels once per feature even if there are multiple segments. For example, the group of islands that make up Hawaii is only labeled once. If there are other features with the same name, they are also labeled.
	Used to set a buffer around the label. When this is set, no labels overlap within the buffer range. The ratio is the fraction of the height or the width of the label rectangle (whichever is smaller) compared to the width of the buffer. A ratio of "0.0" means no buffer. A ratio of "1.0" means that the buffer is twice the size of the label (the label width equals the buffer width). A negative ratio causes the buffer to be smaller than the label. This can be used to allow labels to overlap.
labelPriorities	Used to determine where to place the label around the point. The attribute accepts different weights for each of eight positions around the point. Each position corresponds to the positions as shown below: 2 3 x 4 7 6 5

	In each position, the user can prioritize the importance of that position from 0 upwards. 0 signifies that the label should not be placed in that position. 1 means that this is an acceptable position for the label, and all higher numbers represent lesser priorities for that position. For example, "1,0,1,0,0,0,0,0" means that only the first and third label positions will be taken into account when labeling. In another example, "1,2,3,0,1,0,0,0" means try to label at the first and fifth position; if not, then put the label at the second position; if not, then put it at the third position; if this is not possible, then don't label it. <aims:simpleLabelRenderer field="NAME" labelPriorities="1,2,3,0,1,0,0,0"> Another option is to place a label on top of points rather than around points. To do this, use "LE_PlaceOnTopHorizontal" for the label priority. <aims:simpleLabelRenderer field="NAME" labelPriorities="LE_PlaceOnTopHorizontal">
labelWeight	Used to prioritize the importance of labels. The label weight is usually set to "high_weight" since the labels are more important. This can be set lower if the labels are not as important as the feature.
lineLabelPosition	Determines where on the line to place the label. The following options are available: • PlaceAbove—Place above the line. • PlaceBelow—Place below the line. • PlaceOnTop—Place on the line. • PlaceLeft—Place along the left side of the line (label follows line and is not perpendicular to the line). • PlaceRight—Place along the right side of the line (label follows line and is not perpendicular to the line). • PlaceAboveBelow—Place above or below the line. • PlaceLeftRight—Place at either side of the line. • PlaceInLine—Place anywhere on the line. • PlaceParallel—Place parallel to the line. • PlaceOnTopHorizontal—Place label on top of the line but always horizontal.

- PlaceAtStartAbove—Place label at the start above the line.
- PlaceAtStartOnTop—Place label at the start on top of the line.
- PlaceAtStartBelow—Place label at the start below the line.
- PlaceAtEndAbove—Place label at the end above the line.
- PlaceAtEndOnTop—Place label at the end on top of the line.
- PlaceAtEndBelow—Place label at the end below the line.
- PlaceEitherEndAbove—Place at either end above the line.
- PlaceEitherEndOnTop—Place at either end on top of the line.
- PlaceEitherEndBelow—Place at either end below the line.

```
<VALUEMAPLABELRENDERER lookupfield="ADMN_CLASS" labelfield="ROUTE"
linelabelposition="PlaceOnTop">
```

rotationalAngles	The rotational angles are possible angles that the label can be placed at, relative to the labeled point. By default, labels are always placed horizontally. To rotate a label, a comma-delimited list of up to eight rotational angles can be given and are prioritized from first to last. For example, if the first priority is to place labels at 45 degrees and the second priority is at 30 degrees, the rotational angles attribute would look like this:
	
	`<aims:simpleLabelRenderer field="NAME" rotationalAngles="45,30">`
	
	LabelPriorities always take precedence over *rotationalAngles*. If you find that your labels are not rotating as expected, remove the *labelPriorities* attribute if it is present. Alternatively, you can set all the *labelPriorities* to "0".
	
	`<aims:simpleLabelRenderer field="NAME" rotationalAngles="45,30" >`
	or
	`<aims:simpleLabelRenderer field="NAME" labelPriorities="0,0,0,0,0,0,0,0"`
	`rotationalAngles="45,30" >`

Examples for simpleLabelRenderer:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300"/>
<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:layer layerId="5" visible="true">
    <aims:simpleLabelRenderer field="NAME">
      <aims:textSymbol />
    </aims:simpleLabelRenderer>
  </aims:layer>
</aims:map>
<img src="<%=(myMap)%>" />
```

simpleLineSymbol

Used in: mapping
Parent elements: aims:buffer aims:displayFeatures aims:exact aims:line aims:other aims:polygon aims:range aims:simpleRenderer aims:targetLayer

```
<aims:simpleLineSymbol
    Attributes that set values:

    When using ArcMap Server:
    color ="0,0,0 - 255,255,255"  [0,0,0]
    type ="solid"  [solid]
    width ="0 - NNN"  [0]

    When using Image Server:
    antialiasing ="true | false"  [false]
    capType ="butt | round | square"  [butt]
    color ="0,0,0 - 255,255,255"  [0,0,0]
    joinType ="round | miter | bevel"  [round]
    overlap ="true | false"  [true]
    transparency ="0.0 - 1.0"  [1.0]
    type ="solid | dash | dot | dash_dot | dash_dot_dot"  [solid]
    width ="0 - NNN"  [0]
>

    No Child Elements
</aims:simpleLineSymbol >
```

Description:

Symbol for line features.

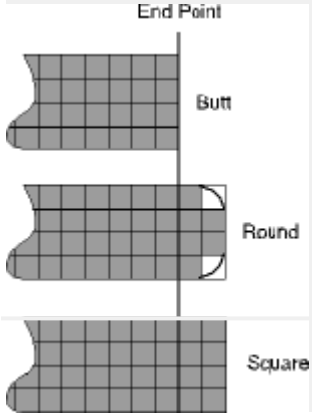
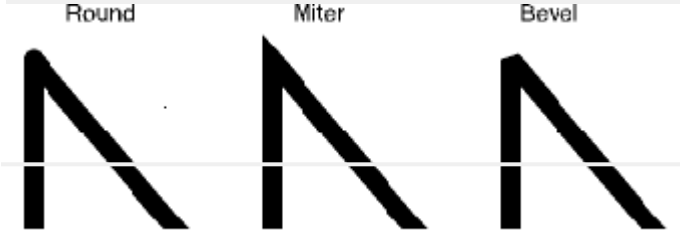
Restrictions:

In ArcMap Image Services, symbol is valid only in acetate layers.

Notes:

None

Attribute descriptions for simpleLineSymbol:

Attribute	Usage
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
capType	Line end style. 
color	Symbol color using RGB values.
joinType	Line join style. 

overlap	Determines if labels can overlap this symbol. When "true", labels can overlap. When "false", labels will not overlap the symbol. If labels are not drawing as expected, check if *overlap* is set to "false" for this symbol or any other symbol in the service.
transparency	Value to set percentage of transparency. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.
type	Line type.
width	Line width in pixels.

Examples for simpleLineSymbol:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300"/>
<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="serviceName"
/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:createLayer layerId="lineLayer" name="line" type="acetate" visible="true">
    <aims:acetateObject units="pixel">
      <aims:line coords="10 150;400 150" >
        <aims:simpleLineSymbol transparency="1.0" type="dash" width="1" capType="round"
joinType="round" color="0,255,0"  />
      </aims:line>
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>
<img src="<%=(myMap)%>" />
```

simpleMarkerSymbol

Used in: mapping
Parent elements: aims:buffer aims:displayFeatures aims:exact aims:line aims:other aims:point aims:polygon aims:range aims:simpleRenderer aims:targetLayer

<aims:simpleMarkerSymbol

Attributes that set values:

When using ArcMap Server:
color ="0,0,0 - 255,255,255" **[0,0,0]**
outline ="0,0,0 - 255,255,255"
type ="circle | triangle | square | cross | star" **[circle]**
width ="1 - NNN" **[3]**

When using Image Server:
antialiasing ="true | false" **[false]**
color ="0,0,0 - 255,255,255" **[0,0,0]**
outline ="0,0,0 - 255,255,255"
overlap ="true | false" **[true]**
shadow ="0,0,0 - 255,255,255"
transparency ="0.0 - 1.0" **[1.0]**
type ="circle | triangle | square | cross | star" **[circle]**
useCentroid ="true | false" **[false]**
width ="1 - NNN" **[3]**
>

No Child Elements
</aims:simpleMarkerSymbol >

Description:

Symbolizes point features using one of the predefined symbol types: circle, triangle, square, cross, or star.

Restrictions:

In ArcMap Image Services, symbol is valid only in acetate layers.

Notes:

None

Attribute descriptions for simpleMarkerSymbol:

Attribute	Usage
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
color	Symbol color using RGB values.
outline	Outline color using RGB values.
	Determines if labels can overlap this symbol. When "true", labels can overlap. When "false", labels will not overlap the symbol. If labels are not drawing as expected, check if *overlap* is set to "false" for this symbol or any other symbol in the service.
shadow	Shadow color using RGB values.
transparency	Value to set percentage of transparency. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.
type	Symbol type.
useCentroid	By default, a marker symbol used on polygon layers draws markers at all polygon vertices. If *useCentroid* is "true", marker is placed in the centroid of the polygon. If multiple polygon parts exist, the marker falls on the part with the biggest area.
width	Symbol width in pixels.

Examples for simpleMarkerSymbol:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />
<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="serviceName"
/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:createLayer layerId="pointLayer" name="point" type="acetate" visible="true">
    <aims:acetateObject units="pixel">
      <aims:point coords="100 100;200 100">
        <aims:simpleMarkerSymbol width="25" type="star" color="255,0,0"
outline="250,250,250" shadow="0,0,0" />
      </aims:point>
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>
<img src="<%=(myMap)%>" />
```

simplePolygonSymbol

Used in: mapping
Parent elements: aims:buffer aims:bufferLayer aims:displayFeatures aims:exact aims:other aims:polygon aims:range aims:simpleRenderer aims:targetLayer

<aims:simplePolygonSymbol

Attributes that set values:

When using ArcMap Server:
boundary ="true | false" **[true]**
boundaryColor ="0,0,0 - 255,255,255" **[0,0,0]**
boundaryType ="solid" **[solid]**
boundaryWidth ="1 - NNN" **[1]**
fillColor ="0,0,0 - 255,255,255" **[0,200,0]**
fillInterval ="2 - NNN" **[6]**
fillType ="solid | bdiagonal | fdiagonal | cross | diagcross | horizontal | vertical" **[solid]**

When using Image Server:
antialiasing ="true | false" **[false]**
boundary ="true | false" **[true]**
boundaryCapType ="butt | round | square" **[butt]**
boundaryColor ="0,0,0 - 255,255,255" **[0,0,0]**
boundaryJoinType ="round | miter | bevel" **[round]**
boundaryTransparency ="0.0 - 1.0" **[1]**
boundaryType ="solid | dash | dot | dash_dot | dash_dot_dot" **[solid]**
boundaryWidth ="1 - NNN" **[1]**
fillColor ="0,0,0 - 255,255,255" **[0,200,0]**
fillInterval ="2 - NNN" **[6]**
fillTransparency ="0.0 - 1.0" **[0]**
fillType ="solid | bdiagonal | fdiagonal | cross | diagcross | horizontal | vertical | gray | lightgray | darkgray" **[solid]**
overlap ="true | false" **[true]**
transparency ="0.0 - 1.0" **[no default]**
>

No Child Elements
</aims:simplePolygonSymbol >

Description:

Symbol for polygon features.

Restrictions:

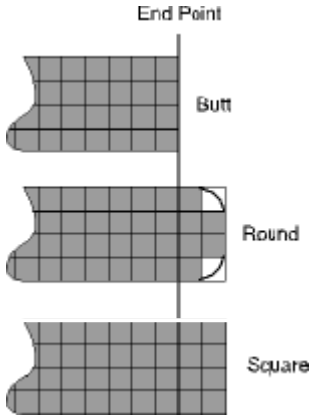
In ArcMap Image Services, symbol is valid only in acetate layers.

Notes:

- For more complex boundary symbols, aims:simpleLineSymbol can be used on polygon layers.
- *Transparency* takes precedence over *fillTransparency* and *boundaryTransparency*.

Attribute descriptions for simplePolygonSymbol:

Attribute	Usage
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
boundary	Turns boundary on or off.
boundaryCapType	Boundary end style.



boundaryColor	Boundary color using RGB values.
boundaryJoinType	Boundary join style. 
	Value to set percentage of transparency for the polygon boundaries. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.
boundaryType	Boundary type.
boundaryWidth	Boundary width.
fillColor	Fill color using RGB values.
fillInterval	Distance between lines for hatch fills.
fillTransparency	Value to set percentage of transparency for the polygon fill. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.
fillType	Symbol fill type.
overlap	Determines if labels can overlap this symbol. When "true", labels can overlap. When "false", labels will not overlap the symbol. If labels are not drawing as expected, check if *overlap* is set to "false" for this symbol or any other symbol in the service.
transparency	Value to set percentage of transparency. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.

Examples for simplePolygonSymbol:

Example 1:

```
<aims:tcpConnection id="mytcpConnection" port="5300" host="myMachine" />
<aims:mapService id="myMapService" connectionId="<%=mytcpConnection%>"
name="serviceName"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:createLayer layerId="idPolygon" name="polygon" type="acetate" visible="true">
    <aims:acetateObject units="pixel">
      <aims:polygon coords="10 110;400 110;400 160;10 160;10 110">
        <aims:simplePolygonSymbol transparency="1.0" fillType="vertical"
fillColor="255,0,255" />
      </aims:polygon>
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>
<img src="<%=(myMap)%>" />
```

simpleRenderer

Used in: mapping
Parent elements: aims:createLayer aims:groupRenderer aims:layer aims:scaleDependentRenderer

<simpleRenderer >
No Attributes

<gradientFillSymbol... /> *[Or]*
<hashLineSymbol... /> *[Or]*
<rasterFillSymbol... /> *[Or]*
<rasterMarkerSymbol... /> *[Or]*
<simpleLineSymbol... /> *[Or]*
<simpleMarkerSymbol... /> *[Or]*
<simplePolygonSymbol... /> *[Or]*
<trueTypeMarkerSymbol... /> *[Or]*

</aims:simpleRenderer >
Bold: Attribute or child element is required.

Description:

Displays features using one symbol.

Restrictions:

- One symbol must be specified. Only one is allowed.
- Not valid with ArcMap Server.

Notes:

None

Examples for simpleRenderer:

Example 1:

```
<aims:tcpConnection id="mytcpConnection" port="5300" host="myMachine" />
<aims:mapService id="myMapService" connectionId="<%=mytcpConnection%>"
name="serviceName"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="350">
<aims:layer layerId="6" visible="true" >
    <aims:simpleRenderer>
        <aims:simpleMarkerSymbol color="0,0,255" type="STAR" width="10"/>
    </aims:simpleRenderer>
</aims:layer>
</aims:map>
<img src="<%=(myMap)%>" />
```

siteSave

Used in: administration
Parent elements: None

```
<aims:siteSave
    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    error ="string"
>

    No Child Elements
</aims:siteSave >
```

Bold: Attribute or child element is required.

Description:

Saves the site configuration.

Restrictions:

None

Notes:

None

Attribute descriptions for siteSave:

Attribute	Usage
connec	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
error	

Examples for siteSave:

Example 1: Saving a site.

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://mymachine"
         role="admin"
      debug="true"
      username="admin"
      password="admin"
/>

<aims:siteSave connectionId="<%=myhttpConnection%>" error="theError" />

<%=theError%>
```

siteUser

Used in: administration
Parent elements: None

```
<aims:siteUser

    Attributes that set values:
    newPassword ="string"
    newUser ="string"

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    error ="string"
>

    No Child Elements
</aims:siteUser >
```

Bold: Attribute or child element is required.

Description:

Sets the username and password for the site.

Restrictions:

None

Notes:

None

Attribute descriptions for siteUser:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
error	Returns an error string if any occurred. Returns "Success" if no error.
newPassword	The new password for the site.
newUser	The new username for the site.

Examples for siteUser:

Example 1:

```
<aims:httpConnection
      id="myhttpConnection"
      url="http://mymachine"
         role="admin"
      debug="true"
      username="myUsername"
      password="myPassword"
/>

<aims:siteUser connectionId="<%=myhttpConnection%>" newUser="newUser"
newPassword="newPassword" error="theError"/>

<%=theError%>
```

subset

Used in: metadata
Parent elements: aims:nestedSearch aims:search

```
<aims:subset
    Attributes that set values:
    docid ="string"
    type ="children | parents | siblings | ancestors | descendants"
>

    No Child Elements
</aims:subset >
```

Description:

Specifies a subset of the documents in the metadata repository to search.

Restrictions:

None

Notes:

By default, all documents in a metadata repository are searched. When aims:subset is included, *name*, *owner*, and *userid* attributes specify the "root document" of the search, and the *type* attribute specifies the relation of the desired documents to the root document.

Attribute descriptions for subset:

Attribute	Usage
	String used to uniquely identify a document. The client used to publish the metadata is responsible for creating the document ID. This ID is automatically assigned when using ArcCatalog. If another client is used, aims:getUUID can be used to request a valid ID. The format for an ID is the following: {HHHHHHHH-HHHH-HHHH-HHHH-HHHHHHHHHHHH} where H is a hexadecimal digit (0–9,a–f,A–F). The ID is limited to 38 characters.
type	Target document's relationship to the source document.

Examples for subset:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300"/>

<aims:search  id="searchResults" connectionId="<%=myConnection%>" service="serviceName"
sort="name" operator="or">

  <aims:subset docId="{9DE86F30-F624-439F-B8C4-71CA3337DB59}" />
  <aims:documentInfo name="Parent" />

</aims:search>
```

targetLayer

Used in: mapping
Parent elements: aims:buffer

```
<aims:targetLayer
    Attributes that set values:
    targetLayerId ="string"
>

    <gradientFillSymbol... />  [Or]
    <hashLineSymbol... />  [Or]
    <rasterFillSymbol... />  [Or]
    <rasterMarkerSymbol... />  [Or]
    <simpleLineSymbol... />  [Or]
    <simpleMarkerSymbol... />  [Or]
    <simplePolygonSymbol... />  [Or]
    <truetypeMarkerSymbol... />  [Or]

</aims:targetLayer >
```

Bold: Attribute or child element is required.

Description:

Defines a target layer for selecting features based on a buffer in the same or different layer in the ArcIMS service.

Restrictions:

- Must refer to an existing layer ID in the map configuration file.
- *Known limitation with the Java Connector*. When using a target layer, the target layer cannot be the same layer as the buffer layer. This limitation is with the Java Connector and not with ArcXML. Using ArcXML, the target and buffer layers can be the same.

Notes:

None

Attribute descriptions for targetLayer:

Attribute	Usage
targetLayerId	Reference to unique layer ID as defined in map configuration file.

Examples for targetLayer:

Example 1:

```
<aims:tcpConnection id="myConnection" host="localhost" port="5300" debug="true" />
<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="SantaClara"/>

<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300"
bufferImage="true" bufferRegion="true">
  <aims:layer layerId="5" visible="true">
    <aims:filter whereExpression="NAME='San Jose'">
      <aims:buffer distance="15" bufferUnits="Miles" performBuffer="true" >
        <aims:bufferLayer>
          <aims:simplePolygonSymbol fillTransparency="1.0" fillType="HORIZONTAL"
boundaryColor="0,0,0" fillColor="0,0,0"/>
        </aims:bufferLayer>
        <aims:targetLayer targetLayerId="4">
          <aims:simpleMarkerSymbol type="STAR" width="20" color="0,0,255" />
        </aims:targetLayer>
      </aims:buffer>
    </aims:filter>
  </aims:layer>
</aims:map>
<img src="<%=(myMap)%>" />
```

tcpConnection

Used in: mapping metadata administration
Parent elements: None

```
<aims:tcpConnection

    Attributes that set values:
    host ="string"
    port ="string"
    debug ="true | false"  [false]
    password ="string"  [null]
    role ="user | admin"  [user]
    username ="string"  [null]

    Attributes that return values:
    id ="object"
    ping ="OK | FAIL | FIRST_LOGIN"
>

    No Child Elements
</aims:tcpConnection >
```

Bold: Attribute or child element is required.

Description:

Creates a TCP connection proxy to use with aims:map.

Restrictions:

When *role* is "admin", the attribute *debug* can only be set to "false".

Notes:

None

Attribute descriptions for tcpConnection:

Attribute	Usage
debug	Shows debug messages in servlet's log file.
host	The ArcIMS Application Server host name.
id	Returns ConnectionProxy object.
password	Password when service authentication is used.
ping	Returns a string showing the status from pinging the ArcIMS Application Server.
	The ArcIMS Application Server port number to communicate on.
role	Defines the role of the connection. For administration requests, *role* must be set to "admin".
username	Username when service authentication is used.

Examples for tcpConnection:

Example 1:

```
<aims:tcpConnection id="mytcpConnection" port="5300" host="myMachine" />
<aims:mapService id="myMapService" connectionId="<%=mytcpConnection%>"
name="serviceName"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300" />

<img src="<%=(myMap)%>" />
```

text

Used in: mapping
Parent elements: aims:acetateObject

```
<aims:text
    Attributes that set values:
    label ="string"
    x ="double"
    y ="double"
>



</aims:text >
```

Bold: Attribute or child element is required.

Description:

Places text on an acetate layer.

Restrictions:

None

Notes:

Used only in acetate layers. To label features, use aims:textSymbol.

Attribute descriptions for text:

Attribute	Usage
label	Text label.
x	X-coordinate for text placement location.
y	Y-coordinate for text placement location.

Examples for text:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />
<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="serviceName"
/>

<aims:map id="myMap" serviceId="<%=myMapService%>" >
  <aims:createLayer layerId="textLayer" type="acetate" name="text" visible="true" >
    <aims:acetateObject units="pixel">
      <aims:text label="My Map" x="150" y="150">
        <aims:textMarkerSymbol fontSize="20"/>
      </aims:text>
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>

<img src="<%=(myMap)%>" />
```

textMarkerSymbol

Used in: mapping
Parent elements: aims:text

<aims:textMarkerSymbol

Attributes that set values:

When using ArcMap Server:
blockout ="0,0,0 - 255,255,255"
font ="Any system font" **[Arial]**
fontColor ="0,0,0 - 255,255,255" **[0,0,0]**
fontSize ="1 - NNN" **[10]**
fontStyle ="regular | bold | italic | underline | bolditalic" **[regular]**
halignment ="left | center | right" **[right]**
outline ="0,0,0 - 255,255,255"
valignment ="top | center | bottom" **[top]**

When using Image Server:
angle ="0.0 - 360.0" **[0]**
antialiasing ="true | false" **[false]**
blockout ="0,0,0 - 255,255,255"
font ="Any system font" **[Arial]**
fontColor ="0,0,0 - 255,255,255" **[0,0,0]**
fontSize ="1 - NNN" **[10]**
fontStyle ="regular | bold | italic | underline | outline | bolditalic" **[regular]**
glowing ="0,0,0 - 255,255,255"
halignment ="left | center | right" **[right]**
interval ="0 - NNN" **[0]**
outline ="0,0,0 - 255,255,255"
overlap ="true | false" **[true]**
printMode ="titlecaps | allupper | alllower | none" **[none]**
transparency ="0.0 - 1.0" **[1.0]**
valignment ="top | center | bottom" **[top]**
>

No Child Elements
</aims:textMarkerSymbol >

Description:

Adds static text to an acetate layer.

Restrictions:

Outline and *glowing* should not be used together; use one or the other.

Notes:

Used only in acetate layers. To label features, use aims:textSymbol.

Attribute descriptions for textMarkerSymbol:

Attribute	Usage
angle	Angle of rotation in degrees moving counterclockwise; 0 degrees is horizontal.
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
blockout	Provides a background behind text. Select color using RGB values.
font	Font name. The name is case sensitive. If font name uses "&", use "&" instead. For example, ESRI Transportation & Civic should be written as ESRI Transportation & Civic.
fontColor	Font color using RGB values.
fontSize	Font size.
fontStyle	Font style.
glowing	Glow color around text using RGB values.
halignment	Horizontal alignment of label compared to label point.
interval	Distance between point and printed label.
	Outline color using RGB values.
overlap	Determines if labels can overlap this symbol. When "true", labels can overlap. When "false", labels will not overlap the symbol. If labels are not drawing as expected, check if *overlap* is set to "false" for this symbol or any other symbol in the service.

printMode	Determines how labels are printed. If "none" is used, no change is made to the label: Welcome to ArcIMS. If "alllower" is used, all letters are lowercase: welcome to arcims. If "allupper" is used, all letters are uppercase: WELCOME TO ARCIMS. If "titlecaps" is used, the first letter of each word in a label is uppercase and everything else is lowercase: Welcome To Arcims.
transparency	Value to set percentage of transparency. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.
valignment	Vertical alignment of label compared to label point.

Examples for textMarkerSymbol:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />
<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="serviceName"
/>

<aims:map id="myMap" serviceId="<%=myMapService%>" >
  <aims:createLayer layerId="textLayer" type="acetate" name="text" visible="true" >
    <aims:acetateObject units="pixel">
      <aims:text label="My Map" x="150" y="150">
        <aims:textMarkerSymbol fontSize="20"/>
      </aims:text>
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>

<img src="<%=(myMap)%>" />
```

textQuery

Used in: metadata
Parent elements: aims:nestedSearch aims:search

```
<aims:textQuery
    Attributes that set values:
    word ="string"
    attribute ="string"
    operators ="and | or"
    tag ="string"
    wordDelimiter ="string"  [,]
    zcode ="string"
>
    No Child Elements
</aims:textQuery >
```

Description:

Searches text within a specified element in metadata documents.

Restrictions:

- Either *tag* or *zcode* must be specified, but not both.
- *Attribute* is valid only with *tag*. It is ignored when *zcode* is used.

Notes:

- An aims:textQuery search can be further restricted by limiting the search to an individual attribute.
- For searching numeric values, see aims:valueQuery.

Attribute descriptions for textQuery:

Attribute	Usage
attribute	Name of an attribute for searching.
operators	Value to use in the search criteria of a text search.
tag	Defines the location of an element inside an XML tree. For example, the element "title" might be inside an XML structure such as <idinfo> <citation> <citeinfo> **<title>...</title>** </citeinfo> </citation> </idinfo> To define "title", the string value for this attribute would be "idinfo/citation/citeinfo/**title**".
word	List of one or more keywords.
wordDelimiter	Delimiter between words listed in the search. The default is a comma (,).
zcode	Numeric code defining semantics from the GEO Attribute Set.

Examples for textQuery:

Example 1:

```
<aims:getSettings id="strPortNumber" value="meta_port_number" />
<aims:getSettings id="strServiceName" value="meta_service_name" />

<aims:tcpConnection id="myConnection" host="<%=(strHostName)%>"
port="<%=(strPortNumber)%>" debug="true" />

<aims:search id="searchResults" connectionId="<%=myConnection%>"
service="<%=(strServiceName)%>" sort="name" sort2="local_area" folderMask="4"
operator="and">
  <aims:textQuery tag="_fulltext" word="river"/>
    <aims:nestedSearch operator="or" >
      <aims:textQuery  tag="metadata/idinfo/keywords/theme/themekey"  word="river,ocean"
operators="or" wordDelimiter="," />
    </aims:nestedSearch>
  </aims:textQuery>
</aims:search>
```

textSymbol

Used in: mapping
Parent elements: aims:exact aims:other aims:range aims:simpleLabelRenderer

```
<aims:textSymbol

    antialiasing ="true | false"  [false]
    blockout ="0,0,0 - 255,255,255"
    font ="Any system font"  [Arial]
    fontColor ="0,0,0 - 255,255,255"  [0,0,0]
    fontSize ="1 - NNN"  [12]
    fontStyle ="regular | bold | italic | underline | outline | bolditalic"  [regular]
    glowing ="0,0,0 - 255,255,255"
    interval ="0 - NNN"  [0]
    outline ="0,0,0 - 255,255,255"
    printMode ="titlecaps | allupper | alllower | none"  [none]
    shadow ="0,0,0 - 255,255,255"
    transparency ="0.0 - 1.0"  [1.0]
>

    No Child Elements
</aims:textSymbol >
```

Description:

Symbol used to label point, line, and polygon layers.

Restrictions:

- Not valid with ArcMap Server.
- *Outline* and *glowing* should not be used together; use one or the other.

Notes:

Aims:textSymbol is used to label features in layers. To add text to an acetate layer, use aims:text.

Attribute descriptions for textSymbol:

Attribute	Usage
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
blockout	Provides a background behind text. Select color using RGB values.
font	"&" instead. For example, ESRI Transportation & Civic should be written as ESRI Transportation & Civic.
fontColor	Font color using RGB values.
fontSize	Font size.
fontStyle	Font style.
glowing	Glow color around text using RGB values.
interval	
outline	Outline color using RGB values.
printMode	Determines how labels are printed. If "none" is used, no change is made to the label: Welcome to ArcIMS. If "alllower" is used, all letters are lowercase: welcome to arcims. If "allupper" is used, all letters are uppercase: WELCOME TO ARCIMS. If "titlecaps" is used, the first letter of each word in a label is uppercase and everything else is lowercase: Welcome To Arcims.
shadow	Shadow color using RGB values.
transparency	Value to set percentage of transparency. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.

Examples for textSymbol:

Example 1:

```
<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:layer layerId="0" visible="true">
    <aims:simpleLabelRenderer field="AREA">
      <aims:textSymbol transparency="0.8" printMode="titlecaps" antialiasing="true"
font="Courier New" fontStyle="bolditalic" fontSize="12" glowing="255,255,255"
shadow="255,255,255" fontColor="0,255,100" blockout="124,124,124" interval="3" />
    </aims:simpleLabelRenderer>
  </aims:layer>
</aims:map>
<img src="<%=(myMap)%>" />
```

trueTypeMarkerSymbol

Used in: mapping
Parent elements: aims:buffer aims:displayFeatures aims:exact aims:line aims:other aims:point aims:polygon aims:range aims:simpleRenderer aims:targetLayer

<aims:trueTypeMarkerSymbol

Attributes that set values:

When using ArcMap Server:
character ="32 - 65535"
angle ="0.0 - 360.0" **[0]**
font ="Any system font" **[Arial]**
fontColor ="0,0,0 - 255,255,255" **[0,0,0]**
fontSize ="1 - NNN" **[12]**
fontStyle ="regular | bold | italic | underline | outline | bolditalic" **[regular]**
outline ="0,0,0 - 255,255,255"

When using Image Server:
character ="32 - 65535"
angle ="0.0 - 360.0" **[0]**
angleField ="*string*"
antialiasing ="true | false" **[false]**
font ="Any system font" **[Arial]**
fontColor ="0,0,0 - 255,255,255" **[0,0,0]**
fontSize ="1 - NNN" **[12]**
fontStyle ="regular | bold | italic | underline | outline | bolditalic" **[regular]**
glowing ="0,0,0 - 255,255,255"
outline ="0,0,0 - 255,255,255"
overlap ="true | false" **[true]**
rotateMethod ="geographic | arithmetic | mod_arithmetic" **[mod_arithmetic]**
shadow ="0,0,0 - 255,255,255"
transparency ="0.0 - 1.0" **[1.0]**
useCentroid ="true | false" **[false]**
>

No Child Elements
</aims:trueTypeMarkerSymbol >

Bold: Attribute or child element is required.

Description:

Symbolizes point features using TrueType symbols.

Restrictions:

- *Outline* and *glowing* should not be used together; use one or the other.
- In ArcMap Image Services, symbol is valid only in acetate layers.

Notes:

- The angle for a symbol can be user defined with *angle*. Angles can also be read from a field in a database table using *angleField*.
- If the attributes *angle* and *angleField* are both present, *angle* takes precedence, and *angleField* is ignored.
- If the symbols are rotating in an unexpected direction, double-check the value for *rotateMethod*.
- If you are using custom TrueType symbols on a UNIX machine, the font must be installed in the following directory:
    - For Solaris:
      /usr/openwin/lib/X11/fonts/TrueType
    - For AIX:
      /usr/lpp/X11/lib/X11/fonts/TrueType
    - For Linux:
      /usr/lib/X11/fonts/TrueType
    - For HP-UX:
      /usr/lib/X11/fonts/ms.st/typefaces

Attribute descriptions for trueTypeMarkerSymbol:

Attribute	Usage
angle	Angle of rotation in degrees.
angleField	The field in the database that contains the angle of rotation for a True Type symbol. The field can be in the layer table or in a joined table. • For shapefiles with no joined tables, the field can be referenced using the short format. field="AREA" • For shapefiles with joined tables, the name of the joined table must be included along with the field. field="JOINEDTABLE.AREA" • joined tables, the field can be referenced using the short format. field="AREA" The full long name can also be used. field="ARCSDENAME.TABLE.AREA" format. field="ARCSDENAME.TABLE.AREA" If both *angle* and *angleField* are used, the attribute *angle* takes precedence.
antialiasing	Used to make edges of labels and symbols smoother. When set to "true", antialiasing is active. Note that the time to generate a map may significantly increase when antialiasing is turned on.
character	Text character ASCII value. The character must be a value between 32 and 255 in a font's character map; characters 0–31 are nonprintable and cannot be used.
font	Font name. The name is case sensitive. If font name uses "&", use "&" instead. For example, ESRI Transportation & Civic should be written as ESRI Transportation & Civic.
fontColor	Font color using RGB values.
fontSize	Font size.
fontStyle	Font style.
glowing	
outline	Outline color using RGB values.

overlap	Determines if labels can overlap this symbol. When "true", labels can overlap. When "false", labels will not overlap the symbol. If labels are not drawing as expected, check if *overlap* is set to "false" for this symbol or any other symbol in the service.
rotateMethod	Three methods of calculating angles are available and apply to both *angle* and *anglefield*: 1. "geographic": An angle of 0 is north, and angles are calculated clockwise from north. 2. "arithmetic": An angle of 0 is east, and angles are calculated counterclockwise from east. 3. "mod_arithmetic": An angle of 0 is north, and angles are calculated counterclockwise from north.
shadow	Shadow color using RGB values.
transparency	Value to set percentage of transparency. 1.0 is 0 percent transparent. 0.0 is 100 percent transparent.
useCentroid	By default, a marker symbol used on polygon layers draws markers at all polygon vertices. If *useCentroid* is "true", marker is placed in the centroid of the polygon. If multiple polygon parts exist, the marker falls on the part with the biggest area.

Examples for trueTypeMarkerSymbol:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />
<aims:mapService id="myMapService" connectionId="<%=myTCPConnection%>" name="serviceName"
/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="300">
  <aims:createLayer layerId="pointLayer" name="point" type="acetate" visible="true">
    <aims:acetateObject units="pixel">
      <aims:point coords="100 100;200 100">
        <aims:trueTypeMarkerSymbol useCentroid="true" transparency="1.0" font="ESRI
Cartography" fontStyle="bold" fontSize="20" character="210" fontColor="255,0,255" />
      </aims:point>
    </aims:acetateObject>
  </aims:createLayer>
</aims:map>
<img src="<%=(myMap)%>" />
```

updated

Used in: metadata
Parent elements: aims:nestedSearch aims:search

```
<aims:updated
    Attributes that set values:
    after ="string"
    before ="string"
>

    No Child Elements
</aims:updated >
```

Description:

Allows metadata searches based on a date.

Restrictions:

At least one attribute must be specified. Both can be used.

Notes:

The format for dates is YYYY-MM-DD hh:mm:ss. The year, month, and date are required. Hours, minutes, and seconds are optional.

Attribute descriptions for updated:

Attribute	Usage
after	Search for documents newer than the date and time given.
before	Search for documents older than the date and time given.

Examples for updated:

Example 1:

```
<aims:search  id="searchResults" connectionId="<%=myConnection%>"
service="PelotonMetadata" sort="name" operator="or">

  <aims:subset docId="{9DE86F30-F624-439F-B8C4-71CA3337DB59}" />
  <aims:documentInfo name="Parent" />
  <aims:updated after="2002-01-24 11:55:47" before="2002-01-24 11:55:47" />

</aims:search>
```

validateConnection

Used in: administration
Parent elements: None

```
<aims:validateConnection

    type ="MAP | METADATA"  [MAP]
    service ="string"

    Attributes that pass objects:
    connectionId ="object"

    Attributes that return values:
    error ="string"
>

    No Child Elements
</aims:validateConnection >
```

Bold: Attribute or child element is required.

Description:

Validate communication with a defined service connection.

Restrictions:

None

Notes:

The service information needs to be set in either aims:mapService or in the *service* attribute of aims:validateConnection. If needed for proper validation, the username and password may have to be set if authentication is required.

Attribute descriptions for validateConnection:

Attribute	Usage
connectionId	Holds information about the connection. The connection is created using aims:httpConnection or aims:tcpConnection.
error	Returns an error string if any occurred. Returns "Success" if no error.
service	The name of the service to validate. Setting this attribute overrides the service set in aims:httpConnection or aims:tcpConnection.
type	The type of service defined in the connection.

Examples for validateConnection:

Example 1:

```
      id="mytcpConnection"
      host="mymachine"
      port="5300"
      debug="true"
      username="publish"
      password="publish"
/>

<aims:validateConnection connectionId="<%=mytcpConnection%>" type="METADATA"
service="Metadata" error="myError" />

<%=myError%>
```

valueMapLabelRenderer

Used in: mapping
Parent elements: aims:createLayer aims:groupRenderer aims:layer aims:scaleDependentRenderer

```
<aims:valueMapLabelRenderer

    labelField ="string"
    lookUpField ="string"
    featureWeight ="no_weight | med_weight | high_weight"  [no_weight]
    howManyLabels ="one_label_per_name | one_label_per_shape | one_label_per_part"
    labelBufferRatio ="double"
    labelPriorities ="0,0,0,0,0,0,0,0 - 8,8,8,8,8,8,8,8 | LE_PlaceOnTopHorizontal"  [2,2,1,4,5,3,2,4]
    labelWeight ="no_weight | med_weight | high_weight"  [high_ weight]
    lineLabelPosition ="See table below for values"  [PlaceAbove]
    rotationalAngles ="string"
>

    (m) <exact... />  [Or]
    (m) <range... />  [Or]
    <other... />

</aims:valueMapLabelRenderer >
```

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Used for labeling features. A field is specified for labeling features based on criteria in aims:range or aims:exact.

Restrictions:

- Aims:exact or aims:range is required. Aims:other is optional.
- Not valid with ArcMap Server.

Notes:

- When rotating symbols, the attribute *labelPriorities* always takes precedence over the attribute *rotationalAngles*. If you find that your labels are not rotating as expected when using Image MapServices, remove the *labelPriorities* attribute if it is present.
- If subfields is used in aims:filter for a layer, any fields used for labeling must be included in the subfields list.

Attribute descriptions for valueMapLabelRenderer:

Attribute	Usage
featureWeight	Prioritizes the importance of features. The feature weight determines how important the feature labeled is for the label placement algorithm. If "no_weight" is specified, then the feature has no importance and can be labeled over. If "high_weight" is specified, then the feature has high importance and cannot be labeled over. Giving importance to features increases the complexity of the labeling problem and also the processing time.
howManyLabels	Determines how often a feature is labeled. • "One_label_per_part" labels all parts of a feature in the case of multipart features. For instance, the state of Hawaii consists of several islands, but they are grouped together as one polygon feature. Each island is labeled. • "One_label_per_name" labels once per feature name. When several features share the same name, only one label is shown. For example, if there is more than one polygon with the name "Residential", only one "Residential" polygon is labeled. • "One_label_per_shape" labels once per feature even if there are multiple segments. For example, the group of islands that make up Hawaii is only labeled once. If there are other features with the same name, they are also labeled.
labelBufferRatio	Used to set a buffer around the label. When this is set, no labels overlap within the buffer range. The ratio is the fraction of the height or the width of the label rectangle (whichever is smaller) compared to the width of the buffer. A ratio of "0.0" means no buffer. A ratio of "1.0" means that the buffer is twice the size of the label (the label width equals the buffer width). A negative ratio causes the buffer to be smaller than the label. This can be used to allow labels to overlap.

labelField	Field containing text for labeling features. The field can be in the layer table or in a joined table. • For shapefiles with no joined tables, the field can be referenced using the short format. labelField="AREA" • For shapefiles with joined tables, the name of the joined table must be included along with the field. labelField="JOINEDTABLE.AREA" • For ArcSDE layers without joined tables, the field can be referenced using the short format. labelField="AREA" The full long name can also be used. labelField="ARCSDENAME.TABLE.AREA" • For ArcSDE layers with joined tables, joined fields must be referenced using the full long format. labelField="ARCSDENAME.TABLE.AREA" space: • Using a shapefile with no joined tables. The short format can be used for field names. labelField="CITY STATE_NAME" • Using a shapefile with joined tables. The name of the joined table must be included along with the field names. labelField="JOINEDTABLE.CITY JOINEDTABLE.STATE_NAME" • Using an ArcSDE layer without joined tables. The short format can be used. labelField="CITY STATE_NAME" The full long name can also be used. labelField="ARCSDENAME.TABLE.CITY ARCSDENAME.JOINEDTABLE.STATE_NAME" • Using an ArcSDE layer with joined tables. The full long format must be used. labelField="ARCSDENAME.TABLE.CITY ARCSDENAME.JOINEDTABLE.STATE_NAME"

labelPriorities	Used to determine where to place the label around the point. The attribute accepts different n corresponds to the positions 8 **X** 7 6 5 In each position, the user can prioritize the importance of that position from 0 upwards. 0 signifies that the label should not be placed in that position. 1 means that this is an acceptable position for the label, and all higher numbers represent lesser priorities for that position. For example, "1,0,1,0,0,0,0,0" means that only the first and third label positions will be taken into account when labeling. In another example, "1,2,3,0,1,0,0,0" means try to label at the first and fifth position; if not, then put the label at the second position; if not, then put it at the third position; if this is not possible, then don't label it. <aims:valueMapLabelRenderer lookUpField="TYPE" labelField="NAME" labelPriorities="1,2,3,0,1,0,0,0"> Another option is to place a label on top of points rather than around points. To do this, use "LE_PlaceOnTopHorizontal" for the label priority. <aims:valueMapLabelRenderer lookUpField="TYPE" labelField="NAME" labelPriorities="LE_PlaceOnTopHorizontal">
labelWeight	Used to prioritize the importance of labels. The label weight is usually set to "high_weight" since the labels are more important. This can be set lower if the labels are not as important as the feature.
lineLabelPosition	• PlaceAbove—Place above the line. • PlaceBelow—Place below the line. • PlaceLeft—Place along the left side of the line (label follows line and is not perpendicular to the line).

- PlaceRight—Place along the right side of the line (label follows line and is not perpendicular to the line).
- PlaceAboveBelow—Place above or below the line.
- PlaceLeftRight—Place at either side of the line.
- PlaceInLine—Place anywhere on the line.

- PlaceOnTopHorizontal—Place label on top of the line but always horizontal.
- PlaceAtStartAbove—Place label at the start above the line.

- PlaceAtStartBelow—Place label at the start below the line.
- PlaceAtEndAbove—Place label at the end above the line.
- PlaceAtEndOnTop—Place label at the end on top of the line.
- PlaceAtEndBelow—Place label at the end below the line.
- PlaceEitherEndAbove—Place at either end above the line.

- PlaceEitherEndBelow—Place at either end below the line.

<VALUEMAPLABELRENDERER lookupfield="ADMN_CLASS" labelfield="ROUTE" linelabelposition="PlaceOnTop">

lookUpField	Name of field used to specify ranges for aims:range or exact values for aims:exact.
rotationalAngles	The rotational angles are possible angles that the label can be placed at, relative to the labeled point. By default, labels are always placed horizontally. To rotate a label, a comma-delimited list of up to eight rotational angles can be given and are prioritized from first to last. For example, if the first priority is to place labels at 45 degrees and the second priority is at 30 degrees, the rotational angles attribute would look like this: <aims:valueMapLabelRenderer lookUpField="TYPE" labelField="NAME" rotationalAngles="45,30">

LabelPriorities always take precedence over *rotationalAngles*. If you find that your labels are not rotating as expected, remove the *labelPriorities* attribute if it is present. Alternatively, you can set all the *labelPriorities* to "0".

<aims:valueMapLabelRenderer lookUpField="TYPE" labelField="NAME"
rotationalAngles="45,30" >
or
<aims:valueMapLabelRenderer lookUpField="TYPE" labelField="NAME"
labelPriorities="0,0,0,0,0,0,0" rotationalAngles="45,30" >

Examples for valueMapLabelRenderer:

Example 1:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />
<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="350">
  <aims:layer layerId="6" visible="true" >
    <aims:valueMapLabelRenderer lookUpField="POP" labelField="NAME" >
        <aims:range lower="0" upper="261892" equality="none" label="less than 261892">
            <aims:textSymbol font="Arial" fontSize="12" fontColor="255,0,255" />
        </aims:range>
        <aims:range lower="261893" upper="522070" equality="none" label="261893 to
522070">
            <aims:textSymbol font="Arial" fontSize="16" fontColor="0,255,0" />
        </aims:range>
        <aims:range lower="522071" upper="782249" equality="none" label="522071 than
782249">
            <aims:textSymbol font="Arial" fontSize="20" fontColor="255,0,0" />
        </aims:range>
    </aims:valueMapLabelRenderer>
  </aims:layer>
</aims:map>
<img src="<%=(myMap)%>" />
```

valueMapRenderer

Used in: mapping
Parent elements: aims:createLayer aims:groupRenderer aims:layer aims:scaleDependentRenderer

```
<aims:valueMapRenderer
    Attributes that set values:
    lookUpField ="string"
>

    (m) <exact... />
    <other... />
    (m) <range... />

</aims:valueMapRenderer >
```

Bold: Attribute or child element is required.
(m): Child element can be used multiple times.

Description:

Renders features according to the value in a specified field. Based on this field, a value map can be created to classify data. This is useful when different types of data are stored in the same layer.

Restrictions:

Not valid with ArcMap Server.

Notes:

None

Attribute descriptions for valueMapRenderer:

Attribute	Usage
	Name of field used to specify ranges or exact values.

Examples for valueMapRenderer:

```
<aims:tcpConnection id="myConnection" host="myMachine" port="5300" debug="true" />
<aims:mapService id="myMapService" connectionId="<%=myConnection%>" name="serviceName"/>
<aims:map id="myMap" serviceId="<%=(myMapService)%>" width="500" height="350">
<aims:layer layerId="0" visible="true" >
    <aims:valueMapRenderer lookUpField="TRACT">
        <aims:exact label="Tract#5121" value="5121" >
            <aims:simplePolygonSymbol fillColor="255,0,0" fillTransparency="1.0"/>
        </aims:exact>
        <aims:exact label="Tract#511898" value="511898" >
            <aims:simplePolygonSymbol fillColor="0,255,0" fillTransparency="1.0"/>
        </aims:exact>
        <aims:other label="other">
            <aims:simplePolygonSymbol fillColor="0,0,255" fillTransparency="1.0"/>
        </aims:other>
    </aims:valueMapRenderer>
</aims:layer>
</aims:map>
<img src="<%=(myMap)%>" />
```

valueQuery

Used in: metadata
Parent elements: aims:nestedSearch aims:search

```
<aims:valueQuery
    Attributes that set values:
    attribute ="string"
    equalTo ="string"
    greaterThan ="string"
    greaterThanOrEqualTo ="string"
    lessThan ="string"
    lessThanOrEqualTo ="string"
    notEqualTo ="string"
    tag ="string"
    zcode ="string"
>
    No Child Elements
</aims:valueQuery >
```

Description:

Searches numeric values within a specified element in a metadata document.

Restrictions:

- Either *tag* or *zcode* must be specified, but not both.
- *Attribute* is valid only with *tag*. It is ignored when *zcode* is used.
- One or two of the operator attributes from the following list must be specified: *lessThan*, *greaterThan*, *lessThanOrEqualTo*, *greaterThanOrEqualTo*, *notEqualTo*, or *equalTo*. If two attributes are specified, they are automatically concatenated together with an "and" operator.

Notes:

For searching text values, see aims:textQuery.

Attribute descriptions for valueQuery:

Attribute	Usage
attribute	Name of an attribute for searching.
equalTo	Equal to operator for comparing two values.
greaterThan	Greater than operator for comparing two values.
greaterThanOrEqualTo	Greater than or equal to operator for comparing two values.
lessThan	Less than operator for comparing two values.
lessThanOrEqualTo	Less than or equal to operator for comparing two values.
notEqualTo	Eliminates a particular value from the search.
tag	Defines the location of an element inside an XML tree. For example, the element "title" might be inside an XML structure such as <idinfo> <citation> <citeinfo> **<title>...</title>** </citeinfo> </citation> </idinfo> To define "title", the string value for this attribute would be "idinfo/citation/citeinfo/**title**".
zcode	Numeric code defining semantics from the GEO Attribute Set.

Examples for valueQuery:

Example 1: Searches a specified list of metadata documents.

```
<aims:getSettings id="hostName" value="meta_host_name" />
<aims:getSettings id="portNumber" value="meta_port_number" />
<aims:getSettings id="serviceName" value="meta_service_name" />
<aims:getSettings id="batchSize" value="result_batch_size" />
```

```
<aims:tcpConnection id="myConnection" host="<%= hostName %>" port="<%= portNumber %>"
debug="true" />

<%
String startBatchAt = request.getParameter("startBatchAt"); // where to start in the
collection of results
if (startBatchAt == null) startBatchAt = "0";
%>

<%-- search for datasets containing the word water and having a content type of 002 (ISO
- Downloadable Data) --%>
<aims:search id="results" connectionId="<%= myConnection %>" service="<%= serviceName %>"
sort="local_area" sort2="contenttype" folderMask="4" totalResults="totalResults"
startResult="startResult" numResults="numResults" batchSize="<%= batchSize %>"
startBatchAt="<%= startBatchAt %>">

   <aims:textQuery tag="_fulltext" word="water" />

   <aims:nestedSearch operator="or">
      <aims:valueQuery equalTo="002"
tag="metadata/distInfo/distributor/distorTran/onLineSrc/orDesc" />
   </aims:nestedSearch>

</aims:search>

<%-- loop through the result set --%>
<aims:iterateMetadata iterate="<%= results %>" size="resultsSize">

<%-- Get the dataset's title --%>
<aims:getElement id="theTitle" loadedElements="<%= theElements %>"
value="metadata/dataIdInfo/idCitation/resTitle" />

<P>
Title: <%= theTitle %>

</aims:iterateMetadata>
```

zoom

Used in: mapping
Parent elements: aims:map

```
    Attributes that set values:
    factor ="0 - NNN"  [0]
    x ="double"
    y ="double"
>

    No Child Elements
</aims:zoom >
```

Description:

Zooms to a map by a specified factor or by centering at a specified x,y coordinate.

Restrictions:

None

Notes:

None

Attribute descriptions for zoom:

Attribute	Usage
	Factor for zooming in.
x	X-coordinate for centering map.
y	Y-coordinate for centering map.

Examples for zoom:

Example 1:

```
<aims:tcpConnection id="myTCPConnection" host="myMachine" port="5300" />
<aims:mapService id="myService" connectionId="<%=myTCPConnection%>" />
<aims:map id="myMapURL" serviceId="<%=myService%>" service="world"  >
  <aims:zoom x="-120.0" y="40.0" factor="2" />
</aims:map>
```

zoomFullExtent

Used in: mapping
Parent elements: aims:map

<zoomFullExtent >
No Attributes

No Child Elements
</aims:zoomFullExtent >

Description:

Zooms to the full extent of the map.

Restrictions:

None

Notes:

None

Examples for zoomFullExtent:

Example 1:

```
<aims:mapService id="myService" connectionId="<%=myTCPConnection%>" />
<aims:map id="myMapURL" serviceId="<%=myService%>" service="world"  >
  <aims:zoomFullExtent />
</aims:map>
```

Migrating AppServerLink applications

A

IN THIS CHAPTER

- **AppServerLink applications in ArcIMS 9.x**

AppServerLink was deprecated in ArcIMS 4.0 and replaced with the Java Connector. While your existing AppServerLink applications will continue to function as long as the arcims_appserverlink JAR file is included in your classpath, we recommend that such applications be migrated to the Java Connector. In order to ease this migration process, this chapter highlights AppServerLink bean methods and the corresponding Java Connector methods.

What happens to my AppServerLink applications in ArcIMS 9.x?

ArcIMS 3.1 supported AppServerLink, a JavaBean component that exposed a set of methods and properties to allow communication with the ArcIMS Application Server. AppServerLink was deprecated in ArcIMS 4.0 and replaced by the Java Connector. The arcims_appserverlink.jar will not be installed with ArcIMS 9.x, but applications developed using the AppServerLink will continue to work in ArcIMS 9.x, provided you have arcims_appserverlink.jar in your classpath.

The AppServerLink and the Java Connector can coexist in ArcIMS 9.x, that is, you can continue to run your AppServerLink

The table below shows the AppServerLink bean methods and the corresponding Java Connector methods.

AppServerLink **com.esri.aims.mtier.beans.AppServerLink**	**Java Connector** **com.esri.aims.mtier.io.ConnectionProxy**
setAppServerHost(String newAppServerHost)	setHost(String newHost)
setAppServerPort(int newAppServerPort)	setPort(int newPort)
sendAXLRequest(String serviceName, String request, String customService[1])	send(String request)
	send(String host, String service[2], int port, int connectionType, String request)
	send(String host, String service[2], int port, int connectionType, String username, String password, String request)
getAppServerHost()	getHost()
getAppServerPort()	getPort()
getResponseAXL()	No equivalent method. The send() now returns string.
getServiceName()	getService()
getAXLResponseAsStream(String serviceName, String request, String customService[1])	sendGetStream(String request)

[1]Image is the default customService. Other applicable values are Extract, Query, and Geocode.

[2]A single field in the Java Connector's send method—service—combines two separate fields in AppServerLink's sendAXLRequest—serviceName and customService.

applications and use the Java Connector for new development. Even though this is possible, it is strongly recommended that you migrate your AppServerLink applications to the Java Connector.

Using the information in the table, a sendAXLRequest in AppServerLink such as:

```
sendAXLRequest("SantaClara", "GET_FEATURES <rest
of ArcXML request string>", "Query")
```

would appear as follows in the Java Connector's send request:

```
send("SantaClara&CustomService=Query", 5300, 0,
"GET_FEATURES <rest of ArcXML request string>")
```

For detailed information on ConnectionProxy methods and other available packages, please refer to the ArcIMS—Java Connector Object Model API Specification, located at <ArcIMS Installation Directory>ArcIMS/Documentation/Java_Connector/api_reference.htm.

What's new in the Java Connector?

Some methods in the object model have been updated or added:

- The Map displayFeatures() method can now use a polygon as a spatial filter in addition to point, lines, and envelope.
- The Dataset class has two new methods - setFromLayer and getFromLayer. These methods allow you to more easily add and manipulate dynamic layers based on existing layers in a service.

Some Java Connector bean samples have been added or modified:

- Sample 4.6 - This sample is a new query example where the buffer and target layers are the same.
- Sample 8.2 - This sample is new and shows how to add an ArcMap Image Service.
- Sample 8.5 - This sample has been updated and shows how to generate SVG output.