

**GIS by ESRI®**

# **Customizing ArcIMS**

**ArcIMS™ 3**

**ActiveX® Connector**



**ESRI**

Copyright © 2000 Environmental Systems Research Institute, Inc.

All rights reserved.

Printed in the United States of America.

The information contained in this document is the exclusive property of Environmental Systems Research Institute, Inc. This work is protected under United States copyright law and the copyright laws of the given countries of origin and applicable international laws, treaties and/or conventions. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, or by any information storage or retrieval system, except as expressly permitted in writing by Environmental Systems Research Institute, Inc. All requests should be sent to Attention: Contracts Manager, Environmental Systems Research Institute, Inc., 380 New York Street, Redlands, CA 92373-8100, USA.

The information contained in this document is subject to change without notice.

**U.S. GOVERNMENT RESTRICTED/LIMITED RIGHTS**

Any software, documentation, and/or data delivered hereunder is subject to the terms of the License Agreement. In no event shall the U.S. Government acquire greater than RESTRICTED/LIMITED RIGHTS. At a minimum, use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in FAR §52.227-14 Alternates I, II, and III (JUN 1987); FAR §52.227-19 (JUN 1987) and/or FAR §12.211/12.212 (Commercial Technical Data/Computer Software); and DFARS §252.227-7015 (NOV 1995) (Technical Data) and/or DFARS §227.7202 (Computer Software), as applicable. Contractor/Manufacturer is Environmental Systems Research Institute, Inc., 380 New York Street, Redlands, CA 92373-8100, USA.

ESRI and the ESRI globe logo are trademarks of Environmental Systems Research Institute, Inc., registered in the United States and certain other countries; registration is pending in the European Community. ArcGIS, ArcIMS, ArcSDE, ArcExplorer, GIS by ESRI, and the ArcIMS logo are trademarks and www.esri.com and @esri.com are service marks of Environmental Systems Research Institute, Inc. Netscape and the Netscape N logo are registered trademarks of Netscape Communications Corporation in the United States and other countries. Microsoft and the Windows logo are registered trademarks and the Microsoft Internet Explorer logo is a trademark of Microsoft Corporation. Other companies and products mentioned herein are trademarks or registered trademarks of their respective trademark owners.

# Contents

## Introducing the ActiveX Connector 5

What is the ActiveX Connector?	6
ActiveX Connector samples and template setup	7
Using the ActiveX Connector samples	8
Using the ActiveX Connector template	10
ASP template frames	11
Setting parameters for the map	13
Creating and using the layer list	14
Removing and adding layers	17
Identifying and querying features	18
Creating a toolbar for zooming and panning	19

## ActiveX Connector Object Model 21

ActiveX Connector objects	22
AcetateLayer	28
AddressMatchInputs	36
AddressMatchResults	51
ArcIMSConnector	60
CalloutMarkerSymbol	67
Circle	80
Envelope	83
FeatureLayer	87
Fields	100
Filter	103
GradientFillSymbol	115
GroupRenderer	123
HashLineSymbol	129
ImageLayer	139
ImageWorkspace	145
ims constants	148
Layers	178
Legend	190
LineObject	209
Map	214
NorthArrowObject	232
Parts	246
PointObject	253
Points	258
PolygonObject	263

RasterFillSymbol 269  
RasterMarkerSymbol 276  
RasterShieldSymbol 286  
Recordset 300  
ScaleBarObject 313  
ScaleDependentRenderer 339  
SDEWorkspace 343  
ShapeWorkspace 352  
ShieldSymbol 356  
SimpleLabelRenderer 364  
SimpleLineSymbol 373  
SimpleMarkerSymbol 382  
SimplePolygonSymbol 391  
SimpleRenderer 403  
TableDesc 405  
TextMarkerSymbol 410  
TextObject 426  
TextSymbol 432  
TrueTypeMarkerSymbol 445  
ValueMapLabelRenderer 458  
ValueMapRenderer 476

# Introducing the ActiveX Connector

# 1

## IN THIS CHAPTER

- **What is the ActiveX Connector?**
- **ActiveX Connector samples and template setup**
- **Using the ActiveX Connector samples**
- **Using the ActiveX Connector template**
- **ActiveX Connector template frames**
- **Setting parameters for the map**
- **Creating and using the layer list**
- **Removing and adding layers**
- **Identifying and querying features**
- **Creating a toolbar for zooming and panning**

ESRI® ArcIMS™ 3 software provides a suite of tools allowing you to create very 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 ArcIMS.

*Customizing ArcIMS* is a series of programming reference books that describes the customization of the HTML and Java™ Viewers and the ActiveX® and ColdFusion® Connectors provided with ArcIMS.

This book explains the foundation for customizing an ActiveX client as well as provides a complete reference to the ActiveX Connector Object Model.

This book assumes that you have a working knowledge of Active Server Page (ASP) objects and VBScript and familiarity with JavaScript and HTML.

In this chapter, you are introduced to

- Reasons for customizing ArcIMS with ASP or one of the Visual InterDev environments such as Visual Basic
- The ASP template and samples provided with ArcIMS
- How different mapping functionality is implemented with ASP

# What is the ActiveX Connector?

ArcIMS has three Application Server Connectors. The connectors are used to connect the Web server to the ArcIMS Application Server. The ActiveX Connector works with custom ActiveX clients to provide a quick and easy solution for incorporating maps into current user applications.

## Considerations for choosing the ActiveX Connector

ArcIMS provides four customizable clients—HTML, Java, ActiveX, and ColdFusion.

The ActiveX Connector offers developers the opportunity to add maps to existing ASP or Visual Basic applications. In addition to basic mapping functionality, such as panning and zooming, you can provide tools to make changes to map layers. Some of the ways you can manipulate a layer include:

- Changing the color or style of a layer
- Adding or removing layers from local data sources
- Updating or adding attribute data

All requests are handled on the server-side, which offers two advantages. The page that is generated is a very thin client that is cross-browser compliant. Also, because the processing is on the server-side, your code is not exposed to the web page and the user.

Joins with the ActiveX Connector, using the JoinTables property of the Filter object, only work with ArcSDE layers. If you need to make a join on a DBF, use an ArcXML request and send it through the ArcIMS Connector object. See chapter 2, ‘ActiveX Connector Object Model’, for more details about the Filter object and JoinTables property. Refer to the ArcXML Programmer’s Reference Guide for information on the syntax for the ArcXML request.

The ActiveX connector must be installed on a Windows NT Server, running IIS. See the *ArcIMS 3.0 Installation Guide*, Installing ArcIMS on Windows NT: ArcIMS custom Application Server Connectors section for directions on how to install the ActiveX Connector.

## Samples and template included with ArcIMS

Twelve ASP samples, an ASP template, and a Visual Basic demo are included with ArcIMS.

The ASP samples are simple demonstrations of the basic mapping functionality of the ActiveX Connector that can be used in ASP. The samples can be incorporated as needed into an existing user application.

The ASP template is a framework for using the ActiveX Connector and ASP. Functions, such as zoom, pan, identify, query, add layers, and change symbols, have been incorporated in this template. The template can be a starting point for more complex ASP applications. This chapter focuses on explaining the mapping functionality as implemented in the ASP template.

The Visual Basic sample application offers a good example of how to incorporate the ActiveX Connector with an existing Visual Basic application. This sample Visual Basic project references the Aims\_activex.dll to display image URLs from ArcIMS Image MapServices.

Several samples and a template have been provided with ArcIMS. They demonstrate a variety of functions. A general set of instructions for installing the samples and template is provided below. You should have a working knowledge of creating MapServices to work with the samples and template.

# ActiveX Connector samples and template setup

## Installing the samples and template

To install the ActiveX Connector, see the *ArcIMS 3.0 Installation Guide* for directions.

To install the samples and template for the ActiveX Connector, choose a Custom Installation when installing ArcIMS. On the ArcIMS Components and Installation Directory panel, check Samples and click Options. On the Options panel, check Sample ActiveX Applications and click OK. This installs the samples and template to <ArcIMS installation directory>\Samples\ActiveX. These directories are included under ActiveX:

<ArcIMS installation directory>\Samples\ActiveX\ASP\_Samples—contains a default.asp file and 12 ASP sample files. The default.asp file shows a list of all the sample files.

<ArcIMS installation directory>\Samples\ActiveX\ASPTemplate—contains ASP and HTML files that make up the template. Also included in this directory is an \images directory that contains the GIF files used as icons for the buttons on the template.

<ArcIMS installation directory>\Samples\ActiveX\Axl—contains the map file (.axl) called SanFrancisco.axl. This map file works with the samples and template. You will use this AXL file to create an Image MapService called SanFrancisco (case sensitive).

<ArcIMS installation directory>\Samples\ActiveX\VBDemo—contains a Visual Basic project and three subdirectories called Classes, Renderers, and Symbols.

## Creating virtual directories for the samples and template

Before you can view the samples and template, you must create virtual directories for them.

For the samples, the alias name is asp\_samples. When setting up the alias, browse to the location of the default.asp: <ArcIMS installation directory>\Samples\ActiveX\ASP\_Samples.

For the template, the alias name is ASPTemplate. When setting up the alias, browse to the location of the template: <ArcIMS installation directory>\Samples\ActiveX\ASPTemplate.

## Starting the SanFrancisco MapService

In ArcIMS Administrator, use the SanFrancisco.axl to start an Image MapService named SanFrancisco (case sensitive).

The SanFrancisco.axl uses data from the C:\Program Files\ESRI\ArcIMS3.0\Samples\TutorialData folder, the default installation folder. If you did not accept the default installation location, you must change the SanFrancisco.axl file to look for the sample data in your workspace. The directory attribute of the SHAPEWORKSPACE tag must be changed:

```
<SHAPEWORKSPACE name="shp_ws-60"  
directory="<My Installation Location>\Samples\TutorialData" />
```

The sample ASP files are a series of code samples that demonstrate particular mapping functionality such as panning and zooming or rendering of layers.

In a browser, type [http://<localhost>/asp\\_samples/default.asp](http://<localhost>/asp_samples/default.asp) to open a listing of the 12 sample ASP files.

# Using the ActiveX Connector samples

## **Sample 1: Show map, list of layers**

Shows the map image and the layers list.

## **Sample 2: Change layer visibility, legend**

Controls the visibility of a layer with the layer list. The check box values are on or off. Click Refresh to update the map.

## **Sample 3: Zoom In, Zoom Out, and Pan**

Change the extent of the map using zoom in, zoom out, or pan. Click one of the choices and click on the map.

## **Sample 4: Change layer's order**

Changes the ordering of the layers by making a layer active and then clicking the Up or Down button. Click Refresh to update the map.

## **Sample 5: Change marker symbol properties**

Sets the size, type, color, outline color, and shadow for a layer. Click Apply to see the changes on the map. The layer must have at least one SimpleRenderer with a SimpleMarkerSymbol.

## **Sample 6: Change polygon symbol properties**

Sets the color, style, and interval of the fill and color, width, and style of the outline for a layer. Click Apply to see the changes on the map. The layer must have at least one SimpleRenderer with a SimplePolygonSymbol.

### **Sample 7: Select and highlight**

Selects and highlights a feature that is clicked.

### **Sample 8: Make spatial query**

Specifies an area to make a selection. Click Refresh to see the selection on the map.

### **Sample 9: Create acetate layer**

Sets the properties of the north arrow and displays it on the acetate layer of the map.

### **Sample 10: Query attribute data**

Queries attribute data on the active layer. Type a where clause and click Find.

### **Sample 11: Identify**

Displays feature attribute data based on an active layer when you click on features in the map.

### **Sample 12: Buffer**

Creates a buffer around a feature and selects features within a set buffer distance. Choose a buffer distance and click Create buffer.

## Using the ActiveX Connector template

In a browser, type `http://<localhost>/ASPTemplate/index.htm` to open the template.

The template shows the foundation of an ASP application built with the ActiveX Connector. You can use the template as a starting point for developing more advanced custom applications. The remainder of this chapter explains the framework of the template.

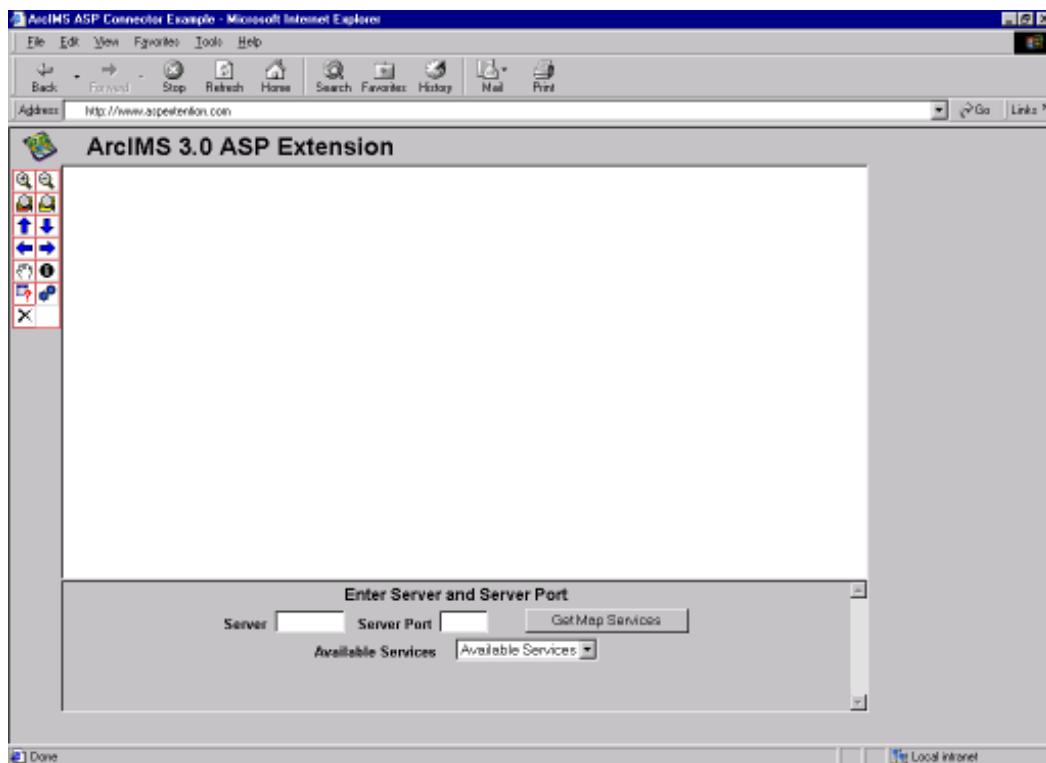
### Setting up the template to use the ActiveX Connector constants

The ActiveX Connector Object Model includes constants that can be used to specify the values of such things as colors or fill styles. See Chapter 2, “ActiveX Connector Object Model”, for a complete listing of the ActiveX Connector constants.

To use these constants, you may need to edit the path in the METADATA TYPE tag. The path should point to the location of the `aims_ActiveX.dll` in the ArcIMS installation directory. The following example shows the path to the ArcIMS installation installed on the D drive:

```
<!-- METADATA TYPE="TypeLib"  
FILE="D:\Program Files\ESRI\ArcIMS3.0\Connectors\ActiveX\aims_ActiveX.dll" -->
```

The remainder of this chapter explains the framework and mapping functionality implemented in the template. When the template first opens, type in the Server name and port number where the ArcIMS Application Server is running to get a list of available MapServices.



# ASP template frames

The ASP template is a web page that contains a set of frames that serve specific mapping purposes. The index.htm file opens the template by defining the frames and specifying the HTML source files for each frame. The six frames that make up the template are TopFrame, ToolFrame, MapFrame, TextFrame, LayerFrame, and BottomFrame. Each of the frames are described below.

## TopFrame and BottomFrame

The TopFrame and BottomFrame are opened by ai\_top.htm and ai\_bottom.htm to hold static pages displaying a gray background for the web page. You can change the contents of these HTML files to change the colors, add your logos, or display text.

## ToolFrame

The ToolFrame holds the toolbar for the template. The ai\_tools.asp file opens the ToolFrame.

For many of the tools, when one is clicked it is selected but not executed. Instead, these tools are executed when the map is clicked or a Submit button is clicked.

A JavaScript function, switchTool(), gets called when a tool is clicked. The switchTool() function passes the value of the tool to MapFrame where it is held in a hidden input. When the map is clicked, a form is submitted to data.asp. Data.asp sets the value of the tool as a server-side variable. Each variable corresponds to a map action or case. The variable is then passed to data.asp and the respective case executed to generate a map. You can find the code for each case in data.asp.

The following code example shows the JavaScript switchtool() function.

```
function switchTool(param){  
if (document.Tools.mapvis.value == "on"){  
document.Tools.functions.value = param;  
parent.MapFrame.mapcontrol.action.value = document.Tools.functions.value;  
}  
}
```

## **MapFrame**

The MapFrame holds the map for the template. The map.htm file provides content for the MapFrame. For many tools, they are set in the ToolFrame but executed in the MapFrame. The purpose of the MapFrame is to hold the map image and the hidden inputs passed from the ToolFrame.

## **TextFrame**

The TextFrame holds different HTML files, depending on the tool currently in use. When the template is first opened, ai\_service.asp opens the TextFrame to show a list of available MapServices. Other files fill the TextFrame to show the results of an identify or the input boxes to create a query.

## **LayerFrame**

The LayerFrame holds the layer list for a MapService. The ai\_layers.htm file provides content for the LayerFrame. The layer list is dynamically generated when a MapService is chosen. The LayerFrame includes four types of functionality: controlling layer visibility, setting layer properties, setting the active layer, and toggling between the layer list and the legend. These functions are described later in this chapter.

## Setting parameters for the map

This section describes setting the map parameters as implemented in the ASP template.

After the template has been initialized, the user must specify an ArcIMS server name and port number to obtain a list of MapServices to choose from. The server name and port are passed through the setmap form and held in variables after the form has been submitted. A list of all the services are added to the Available Services drop down list. This drop down list is populated by using the GetServiceNames() method from the ArcIMSCConnector object to return a list of all the available MapServices for that server.

By clicking on a MapService from the dropdown list, the setmap form (holding the server name, server port, and MapService name) is submitted to the data.asp file. When data.asp is submitted the URL looks like <http://localhost/data.asp?source=Map>. The last portion of the URL tells what case is to be run. The URL in this example runs a case called Map.

Case Map sets all of the parameters for the map. The width and height of the map and the North arrow are all parameters of the map. After the parameters are defined, an ASP Session object is created and set to the Map object. The Session object is used to maintain the current ‘state’ of the user’s copy of the ASP application. For example, each time the user makes a request to the application, the Session object ‘remembers’ the current map extent as well as any symbology changes the user may have made to the MapService’s layers since the web site was first visited by that user. The Session object will be destroyed, or go out of scope, after 20 minutes of inactivity, at which time the MapService will be reinitialized for that user. The use of Session objects is efficient because the server objects are reused instead of recreated.

After the map image has been created based on the parameters, the map is referenced with the GetImageAsURL method. This method returns a string containing the URL that points to the map image. The template places the map image in the MapFrame.

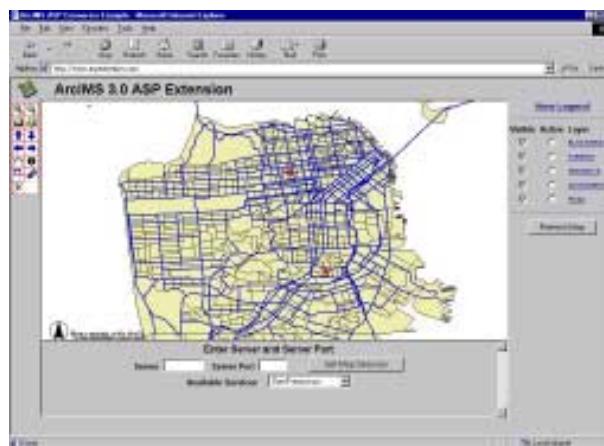
The following code example shows how a URL is passed to retrieve the map image.

<http://localhost/data.asp?source=Map>

Server sets variable to the value of source:

Set vSource = Request.QueryString("source")

Server side variable vSource = Map.



# Creating and using the layer list

This section describes the layer list and legend functionality as implemented in the ASP template.

## Creating the layer list

The LayerFrame is populated based on which MapService is selected. When selecting a MapService, the case LayerList is called to reload the LayerFrame.

Case LayerList returns a list of all layers associated with a MapService. Every time a new MapService gets displayed or layers are added or removed, case LayerList gets called and the layer list is refreshed.

The following code example shows how the layer list is refreshed.

```
<script language="javascript">
document.location = "data.asp?source=LayerList"
</script>
```

Not only does this set the location of this page to data.asp, but it also sets source equal to LayerList.

The following code shows how the layer list is generated.

```
http://localhost/data.asp?source=LayerList
```

Server sets variable to the value of source:

```
Set vSource = Request.QueryString("source")
```

Server side variable vSource = LayerList.

The layer list allows users to change layer visibility, set the active layer, change layer properties, and toggle between the layer list and a more cartographically pleasing legend. These functions are described below.

## Setting the layer visibility

The LayerList case checks each layer's visibility and places a check box that is checked for visible and unchecked for not visible.

When a user clicks one of the visibility check boxes, a JavaScript function, setvis(), is called. The setvis() function tests for the value of the check box and then passes the value to a hidden input. A set of values based on the layer number is passed through another URL string to data.asp. The URL string contains a parameter that sets action = visible. The server parses the URL to find a variable called vAction. If vAction is equal to visible, case Visible is executed. The URL also passes a list of layers and their visibility status to the server. The case Map will then go through and turn layers on or off.

The following code example from data.asp shows how the layer visibility is set.

```
case "visible"
dim theLayerlist(70)
For i = 1 to mapimage.Layers.Count - 1
theLayerlist(i) = Request.QueryString("layer" & i)
'Response.Write theLayer(i)
Next
```

```

For i = 1 to mapimage.Layers.Count - 1
if theLayerlist(i) = "checked" then
mapimage.Layers.Item(i).Visible = True
else
if theLayerlist(i) = "unchecked" then
mapimage.Layers.Item(i).Visible = false
end if
end if
Next
mapimage.Refresh

```

Keep in mind that the map does not get updated until the user clicks Refresh Map.

### **Making a layer active**

By clicking a layer's Active radio button, the layer is made active. The layer must be active in order to perform such tasks as identifying features or removing a layer.

When a layer is made active, a JavaScript function, setActive(), gets called. The layer's name is passed to the setActive function and then passed to both the ToolFrame and MapFrame. The ToolFrame enables the identify and remove functions. The MapFrame executes the identify or remove functions.

### **Changing layer properties**

Each layer name in the legend is hyperlinked to layerprop.asp. When the hyperlink takes place the layer name gets passed to layerprop.asp and is assigned to a variable called theLayer. Before the page appears in the TextFrame, the layer gets tested for layer type, feature class, and type of renderer. A specific rendering form is returned to the TextFrame based on the results of the test. The rendering forms contain controls that handle different ways the layer may be rendered. Once the rendering controls have been selected, clicking the Apply Rendering button executes the appropriate JavaScript function for the renderer being used. The function passes the rendering values to hidden inputs in the MapFrame and submits the MapFrame. Data.asp is then executed and the URL holds all of the values for rendering. The URL is passed to the server. The server checks for case Render and places all of the values from the URL into variables. Case Render tests for the type of rendering and passes the values as rendered values. When case Render is finished, an ArcXML request is sent to the ArcIMS Spatial Server. The response is generated on the server and sent back to the client through the ActiveX Connector.

The following code example from data.asp is a portion of the case Render that shows how to change a polygon layer's fillstyle and fillColor.

```
mapimage.Layers.Item(rendlayer).Renderer.Symbol.FillStyle = fillstyle
```

How to change a polygon layer's fillColor.

```
mapimage.Layer.Item(rendlayer).Renderer.Symbol.FillColor = color
```

FillStyle and FillColor are variables that were passed from the rendering form.

## Toggling between the layer list and legend

To view the Legend, the user clicks the Legend hyperlink. The hyperlink's action in data.asp is set to Legend. On the server-side, the case Legend is executed and the legend image is constructed. Server-side HTML is then generated and called legend. The legend image's URL is passed into the HTML page, and the page is displayed.

The following code example from data.asp shows how to create the legend.

```
case "Legend"  
if isObject(Session("mapsession")) Then  
Set mapimage = Session("mapsession")  
dim theLegend  
Set theLegend = mapimage.Legend  
theLegend.Background = 16777215  
theLegend.AutoExtent = True  
mapimage.Refresh
```

The following code places the URL of the image in an image source to display.

```
dim legend  
legend = "<body bgcolor='white' vlink='navy' alink='navy' link='navy'>" & vbCrLf  
legend = legend & "<center>" & vbCrLf  
legend = legend & "<table border='0'>" & vbCrLf  
legend = legend & "<tr><td align='center'>"  
legend = legend & "<font face='verdana' size='2'><a href='data.asp?source=LayerList'>  
<b>View Layer List</b></a></font></td></tr>"  
legend = legend & "</table>"  
legend = legend & "<table border='0'>"  
legend = legend & "<tr><td align='left' bordercolor='black'><img src=''" &  
theLegend.URL & "'></td></tr>"  
legend = legend & "</table>"  
Response.write legend  
end if
```

# Removing and adding layers

This section describes the adding and removing of layers functionality as implemented in the ASP template.

## Removing layers

When the Remove Layer button is clicked, a variable called theLayer, which is a reference to the active layer, is passed to the MapFrame. Case Delete in data.asp is executed, and the layer is removed.

The following code example from data.asp shows how to remove a layer. The variable theLayer is passed.

```
mapimage.Layers.Remove theLayer
```

## Adding layers

Clicking the Add Layer tool opens the addlayer.asp in the TextFrame. Addlayer.asp contains two text input boxes: one for the directory location (layerdirectory) and the other for the actual name of the shapefile (layerfile). When the user clicks the Add Layer tool a JavaScript function, newlayer(), is called to validate that both input boxes were filled. The values are then passed, the MapFrame is submitted, and the AddLayer case is called. Once the variables are set for both directory and file, a ShapeWorkspace object is created. The filename and directory are passed to the ShapeWorkspace Directory property and the ShapeWorkspace Name property. A layer is then created based on the predefined ShapeWorkspace object and added to the layers collection.

The following code example from data.asp shows how the Add Layers button works.

```
case "addlayer"      'Adds a Layer to the mapservice to display on map
Dim ws
Set dir = Request.QueryString("layerdirectory")
Set file = Request.QueryString("layerfile")
Set ws = CreateObject("aims.shapeworkspace")
ws.Directory = dir
ws.Name = file
ws.FeatureClass = imsFeatureClass.imsPolygon
Set lay = mapimage.Layers.Create(ws)
mapimage.Layers.Add lay
mapimage.Refresh
```

# Identifying and querying features

This section describes the identify and query tools as implemented in the ASP template.

## Identifying features

When the user clicks the Identify button, the value of the tool is set to Identify. The layer's name and action values are passed to the MapFrame. When the user clicks on the map, the image's x and y values get passed to the server and the Identify case gets called. The x and y values are converted to real-world coordinates and used to create an envelope.

The following code example from data.asp shows how the coordinates are converted from pixel to real-world coordinates. Note: The factor is a zoom factor that is added or subtracted to create the envelope.

```
factor = (mapimage.Extent.Xmax - mapimage.Extent.Xmin) / 300
set pt = mapimage.ToMapPoint(x,y)
set env = CreateObject("aims.Envelope")
env.XMin = pt.x - factor
env.YMin = pt.y - factor
env.XMax = pt.x + factor
env.YMax = pt.y + factor
```

A recordset is then created based on the previously created envelope. A field's name and value from the recordset are passed into HTML to be displayed in tabular format. The resulting HTML is then passed to the TextFrame for display.

The following code example from data.asp shows how two variables are set: mLayer and mRecordset. The mLayer variable is set to the layer being identified (the active layer). The mRecordset variable is set as that layer's recordset. Through the recordset's filter AddGObject is added based on env or the previous envelope.

```
set mLayer = mapimage.Layers.Item(theLayer)
Set mRecordset = mLayer.Recordset
mRecordset.Filter.AddGObject env
```

## Querying features

When the Query Builder button is clicked, a JavaScript function, Query(), is called. Query() first tests to make sure the active layer is turned on. (If the layer is turned off, the query will execute and zoom to the results, but the results will not be highlighted on the map.) Then queryform.asp gets passed to the TextFrame. Queryform.asp connects to the predefined session called mapsession. From this session queryform.asp communicates with the previously created Map object. Queryform.asp now has access to all the layers in the layers collection and their recordsets. The Query Builder gets the records and then queries for features.

The following code example shows the JavaScript that loads the querybuilder.

```
function Query(){
if (document.Tools.mapvis.value == "on"){
parent.TextFrame.document.location = "queryform.asp"
}}
```

# Creating a toolbar for zooming and panning

This section describes tools for zooming and panning as implemented in the ASP template.

## Creating a Zoom In tool

When the user clicks on the map, the action is sent to the server and the server executes that action.

The following code example from data.asp shows what happens when a user zooms in with the Zoom In tool. The server searches for case ZoomIn and executes the function. The map is centered on the location of the image clicked by the user and then zoomed in by a factor of one. You can change the zoom factor to meet your application's needs.

```
case "ZoomIn"  
mapimage.CenterAt x, y  
mapimage.DoZoom 1  
mapimage.Refresh
```

## Creating a Zoom Out tool

The following code example from data.asp shows the Zoom Out tool, which works in a similar manner to the Zoom In tool except the zoom factor is set to -1.

```
case "ZoomOut"  
mapimage.CenterAt x, y  
mapimage.DoZoom -1  
mapimage.Refresh
```

## Creating a Zoom to Full Extent tool

The following code example from data.asp shows the Full Extent tool. The value for the action equals FullExtent. The server finds Case FullExtent and processes that section of code. The DoZoomToFullExtent method is called to execute the Full Extent button.

```
case "FullExtent"  
mapimage.DoZoomToFullExtent  
mapimage.Refresh
```

## Creating a Zoom to Active Layer tool

The following code example from data.asp shows the Zoom to Active Layer tool. An envelope called zoomin is created based on the extent of the active layer. The envelope then gets passed to the DoZoomToExtent method. The returned map's extent is based on the active layer's extent.

```
case "ZoomActive"  
Set zoomin = mapimage.Layers.Item(theLayer).Envelope  
mapimage.DoZoomToExtent zoomin  
mapimage.Refresh
```

## **Creating Pan and Pan Directional tools**

The following code example from data.asp shows the Pan tool. When a pan is executed, the map will be returned centered on the user's click.

```
case "Pan"  
mapimage.CenterAt x, y  
mapimage.Refresh
```

The following code examples from data.asp show how to use directional Pan tools to pan the map North, South, East, or West.

The case values are PanNorth, PanSouth, PanEast, and PanWest. The DoPan method pans based on a directional constant and specified step (amount of map to pan). The imsDirection constants are listed in the next chapter.

```
case "PanNorth"  
mapimage.DoPan 1, 3  
mapimage.Refresh
```

```
case "PanSouth"  
mapimage.DoPan 5, 3  
mapimage.Refresh
```

```
case "PanWest"  
mapimage.DoPan 7, 3  
mapimage.Refresh
```

```
case "PanEast"  
mapimage.DoPan 3, 3  
mapimage.Refresh
```

# ActiveX Connector Object Model

# 2

## IN THIS CHAPTER

- ActiveX Connector objects
- ActiveX Connector methods, properties, and constants

This chapter describes the objects, methods, properties, and constants of the ActiveX Connector Object Model.

The ActiveX Connector Object Model is highlighted by two key objects that act as ‘gateways’ to the rest of the objects in the model.

The AimsConnector object must be created before any other object can be accessed. By setting its ServerName and ServerPort properties, a connection is established with the ArcIMS Spatial Server. At this point, there is no association with a MapService.

The Map object is the second critical object to be created. Once the Map object is created, the InitMap method must be invoked before interaction with a MapService can take place. The InitMap method has two required arguments: the AimsConnector object and the name of a MapService that will be accessed by the Map.

Once the Map and AimsConnector objects have been instantiated, all other objects in the model can be referenced.

# ActiveX Connector objects

## AcetateLayer

An AcetateLayer object defines a layer that is displayed on top of the map image. ScaleBarObjects and NorthArrowObjects as well as simple graphics such as LineObjects, PointObjects and PolygonObjects are typically added to this layer. A map can contain multiple acetate layers.

## AddressMatchInputs

The AddressMatchInputs object is a property that contains all of the street address and zip code information necessary to successfully geocode an address on a ‘geocodeable’ map FeatureLayer. Typically, the FeatureLayer represents a street network with address attributes. The AddressMatchInputs object is non-createable.

## AddressMatchResults

The AddressMatchResults object is a property of the AddressMatchInputs object and contains the actual address value, address matching Score, and PointObject for each candidate that was successfully returned from a Geocode request. The AddressMatchResults object is non-createable.

## ArcIMSConnector

The ArcIMSConnector object represents a persistent connection to an ArcIMS Spatial Server. This object is used to initialize a Map object and establish a relationship between that Map object and an ArcIMS MapService. The ArcIMSConnector object and the associated Map object are the gateway through which all of the other ActiveX Connector objects are created.

## CalloutMarkerSymbol

The CallOutMarker Symbol object will display a label and a callout box around the label. This symbol can only be used with point features. It can be rendered with special effects like glowing, shadowing, outlining and antialiasing.

## Circle

The Circle object, like the Envelope object, is a spatial representation object that is used to define a spatial filter for features in a FeatureLayer.

## Envelope

The Envelope object represents the rectangular extent of geographic features. Much like the Circle object, it can also be used to define a spatial filter for features in a FeatureLayer. The Envelope object will also represent the entire geographic extent of a FeatureLayer.

## FeatureLayer

The FeatureLayer object represents all of the geographic features in a dataset that are contained in a single MapService layer. The valid feature classes contained in a MapService’s FeatureLayer object are points, lines, and polygons.

## **Fields**

The Fields object is a collection of database table fields contained in a Recordset object.

## **Filter**

The Filter object is a property of the FeatureLayer object. It contains all of the spatial and attribute constraints used to filter out a subset of features from the FeatureLayer. The Filter object is also a property of the Recordset object and will filter out a subset of records from the Recordset.

## **GradientFillSymbol**

The GradientFillSymbol object will fill a polygon feature with a gradual variation of color, ranging from a ‘start’ color’s value to a ‘finish’ color’s value.

## **GroupRenderer**

The GroupRenderer object represents a collection of renderer objects that are used together to display FeatureLayer information with more complex symbology. A FeatureLayer can have a GroupRenderer associated with it that contains any number of different renderer types.

## **HashLineSymbol**

The HashLineSymbol is used to render linear features using two different line styles. The two line styles represent a background and foreground symbol. One use for the HashLineSymbol would be to display railroad features.

## **ImageLayer**

The ImageLayer object represents an image file that is displayed as a layer in a MapService. It is non-creatable.

## **ImageWorkspace**

The ImageWorkspace object provides a mechanism for locating image files that are to be added to a MapService as ImageLayers. The ‘Directory’ property contains the system directory where the image file is located, and the ‘File’ property contains the name of the image file that is the source for the ImageLayer. The new ImageLayer is then created by invoking the ‘Create’ method in the Map object which takes the ImageWorkspace object as the method parameter.

## **Layers**

The Layers object is a property of the Map object. It is a collection of layer objects in a map service. The Layers collection can contain any combination of FeatureLayers, ImageLayers, and ActetateLayers.

## **Legend**

The Legend object represents a ‘Table of Contents’ for the layers contained in a map service. The Legend object is a property of the Map object, and each Map object has one legend. The Legend object is used to create a cartographically pleasing image of a map service’s legend that can be retrieved and placed into a web page or a Windows-based application.

## **LineObject**

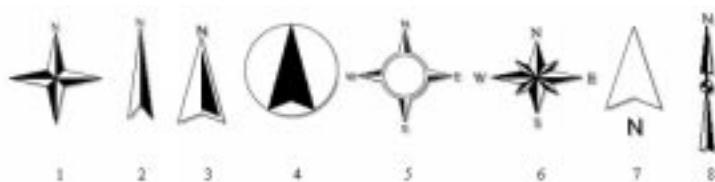
The LineObject represents a linear graphic that can be added to an AcetateLayer. The LineObject can be displayed on an Acetate layer with any of the same symbol properties available to features in a FeatureLayer.

## **Map**

The Map object contains all of the information that can be obtained from a specified map service such as information about each layer, the map extent, and the map units. The Map object is not usable until its ‘InitMap’ method is invoked which requires an ArcIMSConnector and a MapService name. This method will establish a relationship between the Map object and a MapService. Like the ArcIMSConnector, the Map object is a kind of ‘parent’ object, from which all of the other ActiveX Connector objects are created.

## **NorthArrowObject**

The NorthArrowObject is a graphic object that can be added to an AcetateLayer to indicate the orientation of the map display. There are eight different North Arrow styles that can be used.



## **Parts**

The Parts object is a collection of Points objects (see Points) that define the geometry of PolygonObjects and LineObjects. Each ‘part’ of a Parts collection is a Points object collection representing one ‘polygon’ for a PolygonObject or one ‘line segment’ for a LineObject.

## **PointObject**

The PointObject is an XY location object. Generally, a PointObject is added to and rendered in an AcetateLayer. It can also be added to a Filter object and used as a spatial filter. When using the AddressMatching objects, a PointObject is returned from an AddressMatchResults object.

## **Points**

The Points object is a collection of points. The Points object can be used in Line, Polygon, and Parts objects.

## **PolygonObject**

The PolygonObject represents a polygon shape that can be rendered on an AcetateLayer. The PolygonObject has a Parts property, which contains points that represent the coordinates for the polygon boundary.

## **RasterFillSymbol**

The RasterFillSymbol object contains properties that fill a polygon symbol with a specified raster image.

## **RasterMarkerSymbol**

The RasterMarkerSymbol object contains properties that symbolize point features with a specified raster image.

## **RasterShieldSymbol**

A RasterShieldSymbol object is a user-specified image that 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.

## **Recordset**

The Recordset object contains all records for a specified feature layer. Each record in a recordset represents one feature in a feature layer. The Recordset object is non-creatable.

## **ScaleBarObject**

The ScaleBarObject represents the scale bar graphic that indicates the scale of a map. The object is added to and positioned on an AcetateLayer before it can be displayed on a map image. The ScaleBarObject is positioned on a map's AcetateLayer by using its X and Y properties.

## **ScaleDependentRenderer**

The ScaleDependentRenderer object displays specified rendering information at certain scales. A layer can have different renderings depending on the scale threshold. For example, when zoomed out, you can draw a street layer one pixel in width. As you zoom in, you can draw the street layer eight pixels in width and in a different color.

## **SDEWorkspace**

The SDEWorkspace object defines a data source stored in an ArcSDE database. It can be passed to the Create method of the Layers collection to dynamically create new ArcSDE layers for MapServices.

## **ShapeWorkspace**

The ShapeWorkspace object defines a shapefile data source from which shapefiles can be specified and added dynamically to a running map service as a layer.

## **ShieldSymbol**

The ShieldSymbol object is used to add simple roadway shield symbols. The ShieldType property lets a developer choose from five shield options—Interstate, USRoads, Rectangle, Oval, and Mexican.

## **SimpleLabelRenderer**

The SimpleLabelRenderer object is used for labeling features in a feature layer. A field property is used to specify the name of the field from which values will be derived for labeling. Other properties such as FWeight, LWeight, and LabelPriorities are available for considering feature weights, how many features to label, and label rotation and positioning.

## **SimpleLineSymbol**

The SimpleLineSymbol object is used to render line features in a feature layer. Properties such as color, style, and width are used to set a line feature's symbology.

## **SimpleMarkerSymbol**

The SimpleMarkerSymbol object is used to symbolize point features using one of the predefined symbol types: circle, triangle, square, cross, or star.

## **SimplePolygonSymbol**

The SimplePolygonSymbol object is used to symbolize polygon features in a feature layer. Properties such as FillColor, FillStyle, and FillInterval are used to set a polygon feature's symbology.

## **SimpleRenderer**

The SimpleRenderer object is used to display features(point, line, polygon) using one symbol. Layers with a SimpleRenderer will display all features with the same symbology.

## **TableDesc**

The TableDesc object contains all of a recordsets' field information (name, type, length, and precision).

## **TextObject**

The TextObject is used to place text on an acetate layer (created by an AcetateLayer object) to be displayed on the map.

## **TextMarkerSymbol**

The TextMarkerSymbol object is used to set the appearance of text created by the TextObject. Properties such as Font, FontColor, FontStyle, and FontSize alter the way the text is displayed.

## **TextSymbol**

The TextSymbol object is used to label point, line, and polygon features. Properties such as Font, FontColor, FontStyle, and FontSize change the features' labeling.

## **TrueTypeMarkerSymbol**

The TrueTypeMarkerSymbol object defines the characteristics of labels associated with a map layer using a TrueType font. Using a TrueType font allows you to use scalable vector fonts which can be scaled to any size and otherwise transformed more easily than a bitmap font, and with more attractive results.

## **ValueMapLabelRenderer**

A ValueMapLabelRenderer object provides a way to label the features of a map layer by drawing a Symbol for each unique data value or range of data values specified by the Value property. The LookupField property is the name of the field in the layer that stores the text values to use as labels. The Symbol property defines how the text is drawn on the feature. The ValueMapLabelRenderer supports both unique values and ranges.

## **ValueMapRenderer**

The ValueMapRenderer object provides a way to symbolize features of a map layer by drawing a Symbol for each unique data value or range of data values. The LookupField property is the name of the field in the layer that stores the text values to use. Depending on the type of feature, the Symbol property defines how the feature is drawn on the map.

## **AcetateLayer.Add method**

### **Description**

The AcetateLayer.Add method adds an object to the acetate layer.

### **Category**

AcetateLayer

### **Syntax**

AcetateLayer.Add(obj, unit)

### **Arguments**

obj	Object	None	Object to add to acetate layer.
unit	Long	0	Determines how coordinates of object are specified. 0 = pixel units; 1 = database units.

### **Returned Value**

Boolean      True indicates success. False indicates an error.

### **Example**

```
Dim mAcetateLayer
```

```
Dim mPointObject
```

```
Set mAcetateLayer = Server.CreateObject("aims.AcetateLayer")
```

```
mAcetateLayer.visible = true
```

```
Set mPointObject = Server.CreateObject("aims.PointObject")
```

```
mPointObject.x = -117.00
```

```
mPointObject.y = 45.00
```

```
mAcetateLayer.Add mPointObject, 1
```

```
mapimage.Layers.Add mAcetateLayer
```

### **See Also**

[AcetateLayer.Remove](#)

[AcetateLayer.Clear](#)

[AcetateLayer.Item](#)

## **AcetateLayer.Clear method**

### **Description**

The AcetateLayer.Clear method clears all objects from the acetate layer.

### **Syntax**

AcetateLayer.Clear()

### **Arguments**

None

### **Returned Value**

None

### **Example**

Dim mAcetateLayer

Dim mPointObject

```
Set mAcetateLayer = Server.CreateObject("aims.AcetateLayer")
```

```
mAcetateLayer.visible = true
```

```
Set mPointObject = Server.CreateObject("aims.PointObject")
```

```
mPointObject.x = -117.0
```

```
mPointObject.y = 45.0
```

```
mAcetateLayer.Add mPointObject 1
```

```
mapimage.Layers.Add mAcetateLayer
```

```
mAcetateLayer.Clear()
```

### **See Also**

[AcetateLayer.Add](#)

[AcetateLayer.Item](#)

[AcetateLayer.Remove](#)

## **AcetateLayer.Count property**

### **Description**

The AcetateLayer.Count property returns the number of objects in the acetate layer.

### **Syntax**

AcetateLayer.Count

### **Arguments**

None

### **Returned Value**

Long	None	Number of objects in the acetate layer.
------	------	---

### **Example**

```
Dim mAcetateLayer
```

```
Dim mPointObject
```

```
Set mAcetateLayer = Server.CreateObject("aims.AcetateLayer")
mAcetateLayer.visible = true
```

```
Set mPointObject = Server.CreateObject("aims.PointObject")
mPointObject.x = -117.0
mPointObject.y = 45.0
```

```
mAcetateLayer.Add mPointObject 1
mCount = mAcetateLayer.Count
```

```
Response.write mCount
```

### **See Also**

[AcetateLayer.Add](#)

[AcetateLayer.Remove](#)

[AcetateLayer.Clear](#)

## **AcetateLayer.Id property**

### **Description**

The AcetateLayer.Id property returns or sets a string identifier for the acetate layer.

### **Syntax**

AcetateLayer.Id = String

### **Arguments**

None

### **Returned Value**

String            None            Id of the acetate layer.

### **Example**

```
Dim mAcetateLayer  
Set mAcetateLayer = Server.CreateObject("aims.AcetateLayer")  
mAcetateLayer.Id = "ID:1976"  
Response.write mAcetateLayer.Id
```

### **See Also**

AcetateLayer.Name

## **AcetateLayer.Item method**

### **Description**

The AcetateLayer.Item method returns the acetate layer object by index.

### **Syntax**

AcetateLayer.Item(index)

### **Arguments**

index	Long	None	Index to reference of object.
-------	------	------	-------------------------------

### **Returned Value**

Object	None	Acetate object
--------	------	----------------

### **Example**

```
Dim mAcetateLayer
```

```
Dim mPointObject
```

```
Dim mObject
```

```
Set mAcetateLayer = Server.CreateObject("aims.AcetateLayer")
```

```
mAcetateLayer.Visible = true
```

```
Set mPointObject = Server.CreateObject("aims.PointObject")
```

```
mAcetateLayer.Add mPointObject, imsUnit.imsPixel
```

```
Set mObject = mAcetateLayer.Item(1)
```

### **See Also**

[AcetateLayer.Add](#)

[AcetateLayer.Clear](#)

[AcetateLayer.Remove](#)

## **AcetateLayer.Name property**

### **Description**

The AcetateLayer.Name property returns or sets the name of the acetate layer.

### **Syntax**

AcetateLayer.Name

### **Arguments**

None

### **Returned Value**

String            None            Name of the acetate layer.

### **Example**

```
Dim mAcetateLayer  
Set mAcetateLayer = Server.CreateObject("aims.AcetateLayer")  
mAcetateLayer.Name = "AcetateLayer1"  
Response.write mAcetateLayer.Name
```

## **AcetateLayer.Remove method**

### **Description**

The AcetateLayer.Remove method removes the object from the acetate layer by index.

### **Syntax**

AcetateLayer.Add(obj, unit)

### **Arguments**

index	Long	None	Index to reference to the object.
-------	------	------	-----------------------------------

### **Returned Value**

Boolean	True indicates success. False indicates an error.
---------	---

### **Example**

```
Dim mAcetateLayer
```

```
Dim mPointObject
```

```
Set mAcetateLayer = Server.CreateObject("aims.AcetateLayer")
```

```
mAcetateLayer.visible = true
```

```
Set mPointObject = Server.CreateObject("aims.PointObject")
```

```
mPointObject.x = -117.00
```

```
mPointObject.y = 45.00
```

```
mAcetateLayer.Add mPointObject, 1
```

```
mAcetateLayer.Remove 1
```

### **See Also**

[AcetateLayer.Remove](#)

[AcetateLayer.Clear](#)

[AcetateLayer.Item](#)

## **AcetateLayer.Visible property**

### **Description**

The AcetateLayer.Visible property sets the visibility of the acetate layer.

### **Syntax**

AcetateLayer.Visible = True or False

### **Arguments**

None

### **Returned Value**

Boolean      True      True indicates layer is visible. False indicates layer is not visible.

### **Example**

```
Dim mAcetateLayer  
Set mAcetateLayer = Server.CreateObject("aims.AcetateLayer")  
mAcetateLayer.Visible = true
```

## **AddressMatchInputs.AddressMatch method**

### **Description**

The AddressMatchInputs.AddressMatch method executes a geocoding request.

### **Syntax**

AddressMatchInputs.AddressMatch(MaxCandidates, MinScore)

### **Arguments**

MaxCandidates	Long	None	Maximum number of returned candidates.
MinScore	Long	None	Minimum score of returned candidates.

### **Returned Value**

Boolean      True indicates success. False indicates an error.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mAddressMatchInputs  
  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap connect, "IMSMAPService"  
  
Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs  
if mAddressMatchInputs.MoveFirst() then  
    Do  
        if mAddressMatchInputs.Id = "STREET" then  
            mAddressMatchInputs.Value = "195 S. Center"  
        end if  
    Loop  
end if  
mResult = mAddressMatchInputs.AddressMatch(25,60)
```

## **AddressMatchInputs.AddressMatchResults property**

### **Description**

The AddressMatchInputs.AddressMatchResults property contains the result of a geocoding request.

### **Syntax**

AddressMatchInputs.AddressMatchResults

### **Arguments**

None

### **Returned Value**

AddressMatchResults	None	Contains geocoding results.
---------------------	------	-----------------------------

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mAddressMatchInputs  
Dim mAddressMatchResults  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs  
if mAddressMatchInputs.MoveFirst() then  
    if mAddressMatchInputs.Id = "STREET" then  
        mAddressMatchInputs.Value = "195 S Center"  
    end if  
end if  
mResult = mAddressMatchInputs.AddressMatch(25,60)  
Set mAddressMatchResults = mAddressMatchInputs.AddressMatchResults
```

### **See Also**

[AddressMatchResults](#)

## **AddressMatchInputs.ClearValues method**

### **Description**

The AddressMatchInputs.ClearValues method clears the values of all fields.

### **Syntax**

```
AddressMatchInputs.ClearValues()
```

### **Arguments**

None

### **Returned Value**

None

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mAddressMatchInputs
```

```
Dim mAddressMatchResults
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
```

```
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
```

```
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs
```

```
mAddressMatchInputs.ClearValues()
```

## **AddressMatchInputs.Count property**

### **Description**

The AddressMatchInputs.Count property returns the number of fields geocoded.

### **Syntax**

AddressMatchInputs.Count

### **Arguments**

None

### **Returned Value**

Long	None	Number of fields
------	------	------------------

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mAddressMatchInputs
```

```
Dim mAddressMatchResults
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
```

```
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
```

```
mMap.InitMap mArcIMSConnector, "IMSSMapService"
```

```
Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs
```

```
mCount = mAddressMatchInputs.Count
```

```
Response.write CStr(mCount)
```

## **AddressMatchInputs.Description** property

### **Description**

The AddressMatchInputs.Description property contains a brief description of the geocoded items.

### **Syntax**

AddressMatchInputs.Description

### **Arguments**

None

### **Returned Value**

String            None            Brief description.

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mAddressMatchInputs
```

```
Dim mAddressMatchResults
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
```

```
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
```

```
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs
```

```
if mAddressMatchInputs.MoveFirst() then
```

```
    mDesc = mAddressMatchInputs.Description
```

```
end if
```

## **AddressMatchInputs.Id property**

### **Description**

The AddressMatchInputs.Id property returns or sets a string identifier for the geocoded field.

### **Syntax**

AddressMatchInputs.Id

### **Arguments**

None

### **Returned Value**

String            None            Id of the field.

### **Example**

Dim mArcIMSConnector

Dim mMap

Dim mAddressMatchInputs

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs
if mAddressMatchInputs.MoveFirst() then
    id = mAddressMatchInputs.Id
    Response.write id
end if
```

## **AddressMatchInputs.InputType property**

### **Description**

The AddressMatchInputs.InputType property contains the type of the geocoded field.

### **Syntax**

AddressMatchInputs.InputType

### **Arguments**

None

### **Returned Value**

String            None            Type of the field.

### **Example**

Dim mArcIMSConnector

Dim mMap

Dim mAddressMatchInputs

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mAddressMatchInputs = mMap.layers.Item(1).AddressMatchInputs
if mAddressMatchInputs.MoveFirst() then
    mType = mAddressMatchInputs.InputType
    Response.write mType
end if
```

## **AddressMatchInputs.Label property**

### **Description**

The AddressMatchInputs.Label property contains a label for the geocoded items.

### **Syntax**

AddressMatchInputs.Label

### **Arguments**

None

### **Returned Value**

String            None            Label.

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mAddressMatchInputs
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.Server = "IMSServer"
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs
if mAddressMatchInputs.MoveFirst() then
    mLabel = mAddressMatchInputs.Label
    Response.write mLabel
end if
```

## **AddressMatchInputs.MoveFirst method**

### **Description**

The AddressMatchInputs.MoveFirst method sets the internal pointer to the first field.

### **Syntax**

AddressMatchInputs.MoveFirst()

### **Arguments**

None

### **Returned Value**

Boolean      True indicates success. False indicates list is empty.

### **Example**

Dim mArcIMSConnector

Dim mMap

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSSMapService"
```

```
if Not mMap.layers.Item(1).AddressMatchInputs is Nothing then
    mMap.Layers.Item(1).AddressMatchInputs.MoveFirst()
end if
```

## **AddressMatchInputs.MoveLast method**

### **Description**

The AddressMatchInputs.MoveLast method sets the internal pointer to the last field.

### **Syntax**

```
AddressMatchInputs.MoveLast()
```

### **Arguments**

None

### **Returned Value**

Boolean      True indicates success. False indicates list is empty.

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
```

```
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
```

```
mMap.InitMap mArcIMSConnector, "IMSSMapService"
```

```
if Not mMap.Layers.Item(1).AddressMatchInputs is Nothing then
```

```
    mMap.Layers.item(1).AddressMatchInputs.MoveLast()
```

```
end if
```

## **AddressMatchInputs.MoveNext method**

### **Description**

The AddressMatchInputs.MoveNext method sets the internal pointer to the next field.

### **Syntax**

AddressMatchInputs.MoveNext()

### **Arguments**

None

### **Returned Value**

Boolean      True indicates success. False indicates end of list is reached.

### **Example**

Dim mArcIMSConnector

Dim mMap

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSSMapService"
```

```
if Not mMap.Layers.Item(1).AddressMatchInputs is Nothing then
    mMap.Layers.item(1).AddressMatchInputs.MoveNext()
end if
```

## **AddressMatchInputs.MovePrevious method**

### **Description**

The AddressMatchInputs.MovePrevious method sets the internal pointer to the previous field.

### **Syntax**

```
AddressMatchInputs.MovePrevious()
```

### **Arguments**

None

### **Returned Value**

Boolean      True indicates success. False indicates first field is reached.

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
if Not mMap.Layers.Item(1).AddressMatchInputs is Nothing then
    mMap.Layers.item(1).AddressMatchInputs.MovePrevious()
end if
```

## **AddressMatchInputs.Style property**

### **Description**

The AddressMatchInputs.Style contains the name of the geocoding style.

### **Syntax**

AddressMatchInputs.Style

### **Arguments**

None

### **Returned Value**

String            None            Name of the style.

### **Example**

Dim mArcIMSConnector

Dim mMap

Dim mStyle

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
if Not mMap.Layers.Item(1).AddressMatchInputs is Nothing then
```

```
    mStyle = mMap.Layers.item(1).AddressMatchInputs.Style
```

```
    Response.write mStyle
```

```
end if
```

## **AddressMatchInputs.Value property**

### **Description**

The AddressMatchInputs.Value returns or sets the address value to find.

### **Syntax**

AddressMatchInputs.Value

### **Arguments**

None

### **Returned Value**

String            None            Address value to find.

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mAddressMatchInputs
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs
if mAddressMatchInputs.MoveFirst() then
    mAddressMatchInputs.Value = "195 S Center."
end if
```

## **AddressMatchInputs.Width property**

### **Description**

The AddressMatchInputs.Width property returns or sets the width of the field.

### **Syntax**

AddressMatchInputs.Width

### **Arguments**

None

### **Returned Value**

Long            None            Width of the field.

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mAddressMatchInputs
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
```

```
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
```

```
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs
```

```
if mAddressMatchInputs.MoveFirst() then
```

```
    mWidth = mAddressMatchInputs.Width
```

```
    Response.write CStr(mWidth)
```

```
end if
```

## **AddressMatchResults.Count property**

### **Description**

The AddressMatchResults.Count property returns the number of addresses found.

### **Syntax**

AddressMatchResults.Count

### **Arguments**

None

### **Returned Value**

Long            None            Number of addresses.

### **Example**

Dim mArcIMSConnector

Dim mMap

Dim mAddressMatchInputs

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs
if mAddressMatchInputs.MoveFirst() then
```

```
    if mAddressMatchInputs.Id = "STREET" then
```

```
        mAddressMatchInputs.Value = "195 S Center"
```

```
    end if
```

```
end if
```

```
mAddressMatchInputs.AddressMatch(25,60)
```

```
mCount = mAddressMatchInputs.AddressMatchResults.Count
```

```
Response.Write CStr(mCount)
```

## **AddressMatchResults.MoveFirst method**

### **Description**

The AddressMatchResults.MoveFirst method sets the internal pointer to the first address found.

### **Syntax**

```
AddressMatchResults.MoveFirst()
```

### **Arguments**

None

### **Returned Value**

Boolean      True indicates success. False indicates list is empty.

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mAddressMatchInputs
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs
if mAddressMatchInputs.MoveFirst() then
```

```
    if mAddressMatchInputs.Id = "STREET" then
```

```
        mMapAddressMatchInputs.Value = "195 S Center"
```

```
    end if
```

```
end if
```

```
mAddressMatchInputs.AddressMatch(25,60)
```

```
mCount = mAddressMatchInputs.AddressMatchResults.Count
```

```
Response.Write CStr(mCount)
```

## **AddressMatchResults.MoveLast method**

### **Description**

The AddressMatchResults.MoveLast method sets the internal pointer to the last address found.

### **Syntax**

```
AddressMatchResults.MoveLast()
```

### **Arguments**

None

### **Returned Value**

Boolean      True indicates success. False indicates list is empty.

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mAddressMatchInputs
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs
if mAddressMatchInputs.MoveFirst() then
    if mAddressMatchInputs.Id = "STREET" then
        mAddressMatchInputs.Value = "195 S Center"
    end if
end if
mAddressMatchInputs.AddressMatch(25,60)
mAddressMatchInputs.AddressMatchResults.MoveLast()
```

## **AddressMatchResults.MoveNext method**

### **Description**

The AddressMatchResults.MoveNext method sets the internal pointer to the next address in the list.

### **Syntax**

```
AddressMatchResults.MoveNext()
```

### **Arguments**

None

### **Returned Value**

Boolean	True indicates success. False indicates end of list is reached.
---------	---

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mAddressMatchInputs
```

```
Dim mAddressMatchResults
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
```

```
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
```

```
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs
```

```
if mAddressMatchInputs.MoveFirst() then
```

```
    if mAddressMatchInputs.Id = "STREET" then
```

```
        mMap.Value = "195 S Center"
```

```
    end if
```

```
end if
```

```
mAddressMatchInputs.AddressMatch(25,60)
```

```
Set mAddressMatchResults = mAddressMatchInputs.AddressMatchResults
```

```
mAddressMatchResults.MoveNext()
```

## **AddressMatchResults.MovePrevious method**

### **Description**

The AddressMatchResults.MovePrevious method sets the internal pointer to the previous address in the list.

### **Syntax**

AddressMatchResults.MovePrevious()

### **Arguments**

None

### **Returned Value**

Boolean      True indicates success. False indicates first element is reached.

### **Example**

```
Dim mAriMSConnector  
Dim mMap  
Dim mAddressMatchInputs  
Dim mAddressMatchResults  
Set mAriMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mAriMSConnector.ServerName = "IMSServer"  
mAriMSConnector.ServerPort = 5300  
  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mAriMSConnector, "IMSMAPService"  
  
Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs  
if mAddressMatchInputs.MoveFirst() then  
    if mAddressMatchInputs.Id = "STREET" then  
        mAddressMatchInputs.Value = "195 S Center"  
    end if  
end if  
mAddressMatchInputs.AddressMatch(25,60)  
Set mAddressMatchResults = AddressMatchInputs.AddressMatchResults  
mAddressMatchResults.MovePrevious
```

## **AddressMatchResults.Point property**

### **Description**

The AddressMatchResults.Point property contains the address location coordinate.

### **Syntax**

AddressMatchResults.Point

### **Arguments**

None

### **Returned Value**

Point Object              None              Address location coordinate.

### **Example**

```
Dim mArcIMSConnector
Dim mMap
Dim mAddressMatchInputs
Dim mAddressMatchResults
Dim mPointObject
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300

Set mMap = Server.CreateObject( "aims.Map" )
mMap.InitMap mArcIMSConnector, "IMSSMapService"

Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs
if mAddressMatchInputs.MoveFirst() then
    if mAddressMatchInputs.Id = "STREET" then
        mAddressMatchInputs.Value = "380 New York st"
    end if
end if
mResult = mAddressMatchInputs.AddressMatch( 25, 60 )
Set mAddressMatchResults = mAddressMatchInputs.AddressMatchResults
if mAddressMatchResults.MoveFirst() then
    Set mPointObject = mAddressMatchResults.Point

Set al = Server.CreateObject("aims.AcetateLayer")
al.visible = true
Set pt = Server.CreateObject("aims.PointObject")
pt.x = mPointObject.x
```

```
pt.y = mPointObject.y  
al.Add pt, 1  
mMap.Layers.Add al  
mMap.Refresh  
end if
```

## AddressMatchResults.Score property

### Description

The AddressMatchResults.Score property contains address match scoring.

### Syntax

AddressMatchResults.Score

### Arguments

None

### Returned Value

Long            None            Score.

### Example

```
Dim mArcIMSConnector
Dim mMap
Dim mAddressMatchInputs
Dim mAddressMatchResults
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300

Set mMap = Server.CreateObject( "aims.Map" )
mMap.InitMap mArcIMSConnector, "IMSSMapService"
if Not mMap.Layers.Item(1).AddressMatchInputs is Nothing then
    Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs
    if mAddressMatchInputs.MoveFirst() then
        if mAddressMatchInputs.Id = "STREET" then
            mAddressMatchInputs.Value = "380 New York st"
        end if
    end if
    mResult = mAddressMatchInputs.AddressMatch( 25, 60 )
    Set mAddressMatchResults = mAddressMatchInputs.AddressMatchResults
    if mAddressMatchResults.MoveFirst() then
        mScore = mAddressMatchResults.Score
    end if
end if
```

## **AddressMatchResults.Value** property

### **Description**

The AddressMatchResults.Value property returns or sets the geocoding address value.

### **Syntax**

AddressMatchResults.Value

### **Arguments**

None

### **Returned Value**

String            None            Address.

### **Example**

```
Dim mArcIMSConnector
Dim mMap
Dim mAddressMatchInputs
Dim mAddressMatchResults
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300

Set mMap = Server.CreateObject( "aims.Map" )
mResult = mMap.InitMap( mArcIMSConnector, "Redlands" )
if Not mMap.Layers.Item(1).AddressMatchInputs is Nothing then
    Set mAddressMatchInputs = mMap.Layers.Item(1).AddressMatchInputs
    if mAddressMatchInputs.MoveFirst() then
        if mAddressMatchInputs.Id = "STREET" then
            mAddressMatchInputs.Value = "380 New York st"
        end if
    end if
    mResult = mAddressMatchInputs.AddressMatch( 25, 60 )
    Set mAddressMatchResults = mAddressMatchInputs.AddressMatchResults
    if mAddressMatchResults.MoveFirst() then
        mValue = mAddressMatchResults.Value
        Response.Write mValue
    end if
end if
```

## ArcIMSConnector.GetLastError method

### Description

The ArcIMSConnector.GetLastError method returns the last error as a string value.

### Syntax

```
ArcIMSConnector.GetLastError()
```

### Arguments

None

### Returned Value

String	None	The error's description.
--------	------	--------------------------

### Example

```
Dim mArcIMSConnector  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerPort = 5300  
mArcIMSConnector.ServerName = "IMSServer"  
. . .  
mLastError = mArcIMSConnector.GetLastError()  
Response.write mLastError
```

## ArcIMSConnector.GetServiceNames method

### Description

The ArcIMSConnector.GetServiceNames method returns a collection which contains the names of all Image Server services.

### Syntax

ArcIMSConnector.GetServiceNames()

### Arguments

None

### Returned Value

Collection      None      Collection contains names of all Image Server services.

### Example

```
Dim mArcIMSConnector  
Dim mService as Collection  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerPort = 5300  
mArcIMSConnector.ServerName = "IMSServer"  
Set mService = mArcIMSConnector.GetServiceNames()  
  
For i = 1 to mService.Count  
    Response.write mService.Item(i)  
  
Next
```

## ArcIMSConnector.RequestAXL property

### Description

The ArcIMSConnector.RequestAXL property contains the last ArcXML request sent to the ArcIMS Spatial Server. To view this request, choose view source from your browser.

### Syntax

ArcIMSConnector.RequestAXL

### Arguments

None

### Returned Value

String	None	The last ArcXML request to the ArcIMS Spatial Server.
--------	------	---

### Example

```
Dim mArcIMSConnector
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerPort = 5300
mArcIMSConnector.ServerName = "IMSServer"
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSSMapService"
mMap.Width = 500
mMap.Height = 450
mMap.Refresh
mLastRequest = mArcIMSConnector.RequestAXL
Response.write mLastRequest
```

## ArcIMSConnector.ResponseAXL property

### Description

The ArcIMSConnector.ResponseAXL property contains the last ArcXML response sent from the ArcIMS Spatial Server. To view this request, choose view source from your browser.

### Syntax

ArcIMSConnector.ResponseAXL

### Arguments

None

### Returned Value

String            None            The last ArcXML response from the ArcIMS Spatial Server.

### Example

```
Dim mArcIMSConnector  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerPort = 5300  
mArcIMSConnector.ServerName = "IMSServer"  
  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSSMapService"  
mMap.Width = 500  
mMap.Height = 450  
mMap.Refresh  
  
mLastResponse = mArcIMSConnector.ResponseAXL  
Response.Write mLastResponse
```

## ArcIMSConnector.SendAXLRequest method

### Description

The ArcIMSConnector.SendAXLRequest method sends the request to the ArcIMS Spatial Server.

### Syntax

```
ArcIMSConnector.SendAXLRequest(ServiceName, Request)
```

### Arguments

ServiceName	String	None	The name of the requested service.
Request	String	None	ArcXML request.

### Returned Value

String	None	The ArcIMS Spatial Server's response.
--------	------	---------------------------------------

### Example

```
Dim mArcIMSConnector
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerPort = 5300
mArcIMSConnector.ServerName = "IMSServer"
mRequest = "<?xml version=""1.0""?>"
mRequest = mRequest & "<ARCXML version=""1.0.1"">"
mRequest = mRequest & "<REQUEST>"
mRequest = mRequest & "<PROPERTIES>"
mRequest = mRequest & "<ENVELOPE minx=""<MinX></MinX>"" minY=""<MinY></MinY>"" & maxx=""<MaxX></MaxX>"" maxy=""<MaxY></MaxY>"" />"
mRequest = mRequest & "<IMAGESIZE height=""300"" width=""500""/>"
mRequest = mRequest & "<LEGEND font=""Verdana"" width=""650"" titlefontsize=""12"" columns=""4"" cansplit=""true"" splittext=""(continued)"" autoextend=""true"" backgroundcolor=""255,255,255""/>"
mRequest = mRequest & "<DRAW map=""false""/>"
mRequest = mRequest & "</PROPERTIES>"
mRequest = mRequest & "</GET_IMAGE>"
mRequest = mRequest & "</REQUEST>"
mRequest = mRequest & "</ARCXML>"

mArcIMSConnector.SendAxlRequest("IMSSMapService", mRequest)
```

## **ArcIMSConnector.ServerName property**

### **Description**

The ArcIMSConnector.ServerName property contains the ArcIMS Spatial Server name to connect to.

### **Syntax**

ArcIMSConnector.ServerName

### **Arguments**

None

### **Returned Value**

String        “localhost”        Server’s name to connect to.

### **Example**

```
Dim mArcIMSConnector  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerPort = 5300  
mArcIMSConnector.ServerName = "IMSServer"
```

### **Note**

Also refer to samples:

Sample1.asp

## ArcIMSConnector.ServerPort property

### Description

The ArcIMSConnector.ServerPort property contains the server port to connect to.

### Syntax

ArcIMSConnector.ServerPort

### Arguments

None

### Returned Value

Long            5300            Server's port to connect to.

### Example

```
Dim mArcIMSConnector  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "localhost"  
mArcIMSConnector.ServerPort = 5300
```

### See Also

[ArcIMSConnector.ServerName](#)

### Note

Also refer to samples:

[Sample1.asp](#)

## **CalloutMarkerSymbol.Antialiasing property**

### **Description**

The CalloutMarkerSymbol.Antialiasing property turns antialiasing on/off. Antialiasing allows lines to appear smooth by coloring adjacent pixels in such a way as to give the edge of a line a more gradual fade to the background.

### **Syntax**

CalloutMarkerSymbol.Antialiasing

### **Arguments**

None

### **Returned Value**

Boolean      True means that antialiasing is on. False means that it is off. The default value is False.

### **Example**

```
Dim mCalloutMarkerSymbol  
Set mCalloutMarkerSymbol = Server.CreateObject("aims.CalloutMarkerSymbol")  
mCalloutMarkerSymbol.Antialiasing = False
```

## **CalloutMarkerSymbol.BackColor property**

### **Description**

The CalloutMarkerSymbol.BackColor property returns or sets a color constant for the background color of a CalloutMarkerSymbol.

### **Syntax**

CalloutMarkerSymbol.BackColor

### **Arguments**

None

### **Returned Value**

imsColor              Background color constant. The default value is imsWhite (16777215).

### **Example**

```
Dim mCalloutMarkerSymbol  
Set mCalloutMarkerSymbol = Server.CreateObject("aims.CalloutMarkerSymbol")  
mCalloutMarkerSymbol.BackColor = imsColor.imsRed
```

## **CalloutMarkerSymbol.BoundaryColor property**

### **Description**

The CalloutMarkerSymbol.BoundaryColor property returns or sets a color constant for the boundary color of a CalloutMarkerSymbol.

### **Syntax**

CalloutMarkerSymbol.BoundaryColor

### **Arguments**

None

### **Returned Value**

imsColor      Boundary color constant. The default value is imsBlack (0).

### **Example**

```
Dim mCalloutMarkerSymbol  
Set mCalloutMarkerSymbol = Server.CreateObject("aims.CalloutMarkerSymbol")  
mCalloutMarkerSymbol.BoundaryColor = imsColor.imsRed
```

## **CalloutMarkerSymbol.Clone method**

### **Description**

The CalloutMarkerSymbol.Clone method clones the CalloutMarkerSymbol.

### **Syntax**

CalloutMarkerSymbol.Clone()

### **Arguments**

None

### **Returned Value**

CalloutMarkerSymbol      Cloned symbol.

### **Example**

```
Dim mClone  
Dim mCalloutMarkerSymbol  
Set mCalloutMarkerSymbol = Server.CreateObject("aims.CalloutMarkerSymbol")  
Set mClone = mCalloutMarkerSymbol.Clone()
```

## **CalloutMarkerSymbol.Font property**

### **Description**

The CalloutMarkerSymbol.Font property returns or sets the font name for text labels used with a CalloutMarkerSymbol.

### **Syntax**

CalloutMarkerSymbol.Font

### **Arguments**

None

### **Returned Value**

String            Font name. The default value is “default”.

### **Example**

```
Dim mCalloutMarkerSymbol  
Set mCalloutMarkerSymbol = Server.CreateObject("aims.CalloutMarkerSymbol")  
mCalloutMarkerSymbol.Font = "Arial"
```

## **CalloutMarkerSymbol.FontColor property**

### **Description**

The CalloutMarkerSymbol.FontColor property returns or sets a font color constant for the font used with CalloutMarkerSymbol.

### **Syntax**

CalloutMarkerSymbol.Color

### **Arguments**

None

### **Returned Value**

imsColor                  Font color. The default value is imsBlack (0).

### **Example**

```
Dim mCalloutMarkerSymbol  
Set mCalloutMarkerSymbol = Server.CreateObject("aims.CalloutMarkerSymbol")  
mCalloutMarkerSymbol.Color = imsColor.imsRed
```

## **CalloutMarkerSymbol.FontSize property**

### **Description**

The CalloutMarkerSymbol.FontSize property returns or sets the font size for the font used with a CalloutMarkerSymbol.

### **Syntax**

CalloutMarkerSymbol.FontSize

### **Arguments**

None

### **Returned Value**

Long            Font size. The default value is 12.

### **Example**

```
Dim mCalloutMarkerSymbol  
Set mCalloutMarkerSymbol = Server.CreateObject("aims.CalloutMarkerSymbol")  
mCalloutMarkerSymbol.FontSize = 10
```

## **CalloutMarkerSymbol.FontStyle property**

### **Description**

The CalloutMarkerSymbol.FontStyle property returns or sets a font style constant for the font used with a CalloutMarkerSymbol. The possible values are:

- 0 = imsRegular
- 1 = imsBold
- 2 = imsItalic
- 3 = imsUnderline
- 4 = imsOutline

### **Syntax**

CalloutMarkerSymbol.FontStyle

### **Arguments**

None

### **Returned Value**

imsFontStyle      Font style. The default value is imsRegular (0).

### **Example**

```
Dim mCalloutMarkerSymbol  
Set mCalloutMarkerSymbol = Server.CreateObject("aims.CalloutMarkerSymbol")  
mCalloutMarkerSymbol.FontStyle = imsFontStyle.imsBold
```

## **CalloutMarkerSymbol.Glowing property**

### **Description**

The CalloutMarkerSymbol.Glowing property returns or sets the glowing color constant for labels used with a CalloutMarkerSymbol. Glowing is the halo effect around text labels.

### **Syntax**

CalloutMarkerSymbol.Glowing

### **Arguments**

None

### **Returned Value**

imsColor                   Glowing color. The default value is imsBlack (0).

### **Example**

```
Dim mCalloutMarkerSymbol  
Set mCalloutMarkerSymbol = Server.CreateObject("aims.CalloutMarkerSymbol")  
mCalloutMarkerSymbol.Glowing = imsColor imsRed
```

## **CalloutMarkerSymbol.Interval property**

### **Description**

The CalloutMarkerSymbol.Interval property returns or sets the distance between a point and a callout box. A small number will bring the box closer to the point.

### **Syntax**

CalloutMarkerSymbol.Interval

### **Arguments**

None

### **Returned Value**

Double      Distance between point and callout box. The default value is 10.

### **Example**

```
Dim mCalloutMarkerSymbol  
Set mCalloutMarkerSymbol = Server.CreateObject("aims.CalloutMarkerSymbol")  
mCalloutMarkerSymbol.Interval = 5
```

## **CalloutMarkerSymbol.Outline property**

### **Description**

The CalloutMarkerSymbol.Outline property returns or sets the outline color constant for the CalloutMarkerSymbol.

### **Syntax**

CalloutMarkerSymbol.Outline

### **Arguments**

None

### **Returned Value**

imsColor              Outline color constant. The default value is imsBlack (0).

### **Example**

```
Dim mCalloutMarkerSymbol  
Set mCalloutMarkerSymbol = Server.CreateObject("aims.CalloutMarkerSymbol")  
mCalloutMarkerSymbol.Outline = imsColor imsRed
```

## **CalloutMarkerSymbol.Shadow property**

### **Description**

The CalloutMarkerSymbol.Shadow property returns or sets a color constant for the shadow color of a CalloutMarkerSymbol.

### **Syntax**

CalloutMarkerSymbol.Shadow

### **Arguments**

None

### **Returned Value**

imsColor              Shadow color constant. The default value is 0.

### **Example**

```
Dim mCalloutMarkerSymbol  
Set mCalloutMarkerSymbol = Server.CreateObject("aims.CalloutMarkerSymbol")  
mCalloutMarkerSymbol.Shadow = imsColor.imsRed
```

## **CalloutMarkerSymbol.Transparency property**

### **Description**

The CalloutMarkerSymbol.Transparency property returns or sets the transparency coefficient for a CalloutMarkerSymbol. Smaller numbers indicate more transparency.

### **Syntax**

CalloutMarkerSymbol.Transparency

### **Arguments**

None

### **Returned Value**

Double Transparency coefficient. The default value is 1.0.

### **Example**

```
Dim mCalloutMarkerSymbol  
Set mCalloutMarkerSymbol = Server.CreateObject("aims.CalloutMarkerSymbol")  
mCalloutMarkerSymbol.Transparency = 1.0
```

## **Circle.Distance property**

### **Description**

The Circle.Distance property contains the radius of the circle.

### **Syntax**

Circle.Distance

### **Arguments**

None

### **Returned Value**

Double	None	Radius of the circle in database units.
--------	------	---

### **Example**

```
Dim mCircle  
Set mCircle = Server.CreateObject("aims.Circle")  
mCircle.Distance = 0.12
```

## **Circle.X property**

### **Description**

The Circle.X property contains the X coordinate of the center.

### **Syntax**

Circle.X

### **Arguments**

None

### **Returned Value**

Double        None        X coordinate of the center.

### **Example**

```
Dim mCircle  
Set mCircle = Server.CreateObject("aims.Circle")  
mCircle.X = -37.12
```

## **Circle.Y property**

### **Description**

The Circle.Y property contains the Y coordinate of the circle's center.

### **Syntax**

Circle.Y

### **Arguments**

None

### **Returned Value**

Double            None            Y coordinate of the center.

### **Example**

```
Dim mCircle  
Set mCircle = Server.CreateObject("aims.Circle")  
mCircle.Y = 122.67
```

## **Envelope.Xmax property**

### **Description**

The Envelope.Xmax property contains the X maximum coordinate.

### **Syntax**

Envelope.Xmax

### **Arguments**

None

### **Returned Value**

Double        None        X maximum coordinate

### **Example**

```
Dim mEnvelope  
Set mEnvelope = Server.CreateObject("aims.Envelope")  
mEnvelope.Xmax = 12
```

### **Note**

Also refer to samples:

Sample8.asp

Sample12.asp

## **Envelope.Xmin property**

### **Description**

The Envelope.Xmin property contains the X minimum coordinate.

### **Syntax**

Envelope.Xmin

### **Arguments**

None

### **Returned Value**

Double        None        X minimum coordinate.

### **Example**

```
Dim mEnvelope  
Set mEnvelope = Server.CreateObject( "aims.Envelope" )  
mEnvelope.XMin = 7
```

### **Note**

Also refer to samples:

[Sample8.asp](#)  
[Sample12.asp](#)

## **Envelope.Ymax property**

### **Description**

The Envelope.Ymax property contains the Y maximum coordinate.

### **Syntax**

Envelope.Ymax

### **Arguments**

None

### **Returned Value**

Double            None            Y maximum coordinate.

### **Example**

```
Dim mEnvelope  
Set mEnvelope = Server.CreateObject("aims.Envelope")  
mEnvelope.XMax = 60
```

### **Note**

Also refer to samples:

Sample8.asp

Sample12.asp

## **Envelope.Ymin property**

### **Description**

The Envelope.Ymin property contains the Y minimum coordinate.

### **Syntax**

Envelope.Ymin

### **Arguments**

None

### **Returned Value**

Double            None            Y minimum coordinate.

### **Example**

```
Dim mEnvelope  
Set mEnvelope = Server.CreateObject( "aims.Envelope" )  
mEnvelope.YMin = 13
```

### **Note**

Also refer to samples:

[Sample8.asp](#)  
[Sample12.asp](#)

## **FeatureLayer.AddressMatchInputs property**

### **Description**

The FeatureLayer.AddressMatchInputs property contains data necessary for geocoding. It is only for geocodable layers.

### **Syntax**

FeatureLayer.AddressMatchInputs

### **Arguments**

None

### **Returned Value**

AddressMatchInputs      None      AddressMatchInputs object.

### **Example**

Dim mArcIMSConnector

Dim mMap

Dim mAddress

```
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject( "aims.Map" )
mMap.InitMap mArcIMSConnector, "IMSSMapService"
if Not mMap.Layers.Item(1).AddressMatchInputs is Nothing then
    Set mAddress = mMap.Layers.Item(1).AddressMatchInputs
end if
```

## FeatureLayer.Clone method

### Description

The FeatureLayer.Clone method creates the clone of a specific layer.

### Syntax

```
FeatureLayer.Clone()
```

### Arguments

None

### Returned Value

FeatureLayer	None	Clone of the featureclass layer.
--------------	------	----------------------------------

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim mClone  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set mClone = mMap.Layers.Item(1).Clone()  
mClone.Renderer.Item(1).Symbol.Color = 255  
mMap.Layers.Add mClone  
mMap.Refresh
```

### Note

Also refer to samples:

- Sample7.asp
- Sample8.asp
- Sample12.asp

## **FeatureLayer.FeatureClass property**

### **Description**

The FeatureLayer.FeatureClass property contains the type of a featureclass layer (point, line, or polygon).

### **Syntax**

FeatureLayer.FeatureClass

### **Arguments**

None

### **Returned Value**

String            None            Type of featureclass layer (point, line, or polygon).

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
MArcIMSConnector.ServerName = "IMSServer"  
MArcIMSConnector.ServerPort = 5300  
  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
mFeatureClass = mMap.Layers.Item(1).FeatureClass  
Response.write mFeatureClass
```

## FeatureLayer.Filter property

### Description

The FeatureLayer.Filter property contains the spatial filter of the layer. FeatureLayer filter affects only the output image of the layer.

### Syntax

FeatureLayer.Filter

### Arguments

None

### Returned Value

Filter      None      Spatial filter of the layer.

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim mFilter  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "World"  
Set mFilter = mMap.Layers.Item(1).Filter
```

### See Also

Filter  
Recordset.Filter

### Note

Also refer to samples:

Sample7.asp  
Sample8.asp  
Sample12.asp

## **FeatureLayer.Id property**

### **Description**

The FeatureLayer.Id property returns or sets a string identifier for the layer.

### **Syntax**

FeatureLayer.Id

### **Arguments**

None

### **Returned Value**

String            None            Id of the layer.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
mId = mMap.Layers.Item(1).Id  
Response.write mId
```

## **FeatureLayer.MaxScale property**

### **Description**

The FeatureLayer.MaxScale property contains the maximum scale to display the layer.

### **Syntax**

FeatureLayer.MaxScale

### **Arguments**

None

### **Returned Value**

Double            0            Maximum scale value.

### **See Also**

[FeatureLayer.MinScale](#)

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
mMap.Layers.Item(1).MaxScale = 50000  
mMap.Refresh
```

## **FeatureLayer.MinScale property**

### **Description**

The FeatureLayer.MinScale property contains the minimum scale to display the layer.

### **Syntax**

FeatureLayer.MinScale

### **Arguments**

None

### **Returned Value**

Double            0            Minimum scale value.

### **See Also**

FeatureLayer.MaxScale

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
mMap.Layers.Item(1).MinScale = 25000  
mMap.Refresh
```

## **FeatureLayer.Name property**

### **Description**

The FeatureLayer.Name property returns or sets the name of the layer.

### **Syntax**

FeatureLayer.Name

### **Arguments**

None

### **Returned Value**

String            None            Name of the layer.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSSMapService"  
 mName = mMap.Layers.Item(1).Name  
Response.write mName
```

## **FeatureLayer.Recordset property**

### **Description**

The FeatureLayer.Recordset property contains the layer's record sets. Recordset is not Nothing only for layers which have a specified Id.

### **Syntax**

FeatureLayer.Recordset

### **Arguments**

None

### **Returned Value**

Recordset      None      Recordset object.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mRecordset  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set env = Server.CreateObject("aims.Envelope")  
env.Xmax = 25  
env.Xmin = -25  
env.Ymax = 25  
env.Ymin = -25  
mMap.Layers.Item(1).Recordset.Envelope = env  
mMap.Layers.Item(1).Recordset.MoveFirst  
For i = 1 to mMap.Layers.Item(1).Recordset.Fields.Count  
Response.write mMap.Layers.Item(1).Recordset.Fields.FieldValueAsString(i)  
Next
```

### **See Also**

Recordset

### **Note**

Also refer to samples:

Sample10.asp

Sample11.asp

## FeatureLayer.Renderer property

### Description

The FeatureLayer.Renderer returns or sets the layer's renderer as a Renderer Object.

### Syntax

FeatureLayer.Renderer

### Arguments

None

### Returned Value

Object                  None                  Layer's renderer.

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim mRenderer  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
mMap.Layers.Item(1).Renderer.Item(1).Symbol.Color = 255  
mMap.Refresh
```

### See Also

[SimpleRenderer](#)  
[SimpleLabelRenderer](#)  
[GroupRenderer](#)  
[ScaleDependentRenderer](#)  
[ValueMapRenderer](#)  
[ValueMapLabelRenderer](#)

### Note

Also refer to samples:

[Sample5.asp](#)  
[Sample6.asp](#)  
[Sample7.asp](#)  
[Sample8.asp](#)  
[Sample12.asp](#)

## **FeatureLayer.Visible property**

### **Description**

The FeatureLayer.Visible sets the visibility of the layer.

### **Syntax**

FeatureLayer.Visible

### **Arguments**

None

### **Returned Value**

Boolean      True indicates layer is visible. False indicates layer is invisible.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSSMapService"  
mMap.Layers.Item(1).Visible = True  
mMap.Refresh
```

### **Note**

Also refer to samples:

- Sample2.asp
- Sample5.asp
- Sample6.asp
- Sample8.asp

## FeatureLayer.Workspace property

### Description

The FeatureLayer.Workspace property exists only for a layer created through the method Layers.Create(), otherwise this property is Nothing. There are two types of workspace objects for featureclass layers: ShapeWorkspace and SDEWorkspace.

### Syntax

FeatureLayer.Workspace

### Arguments

None

### Returned Value

Object                  None                  Workspace object.

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim mShapeWorkspace  
Dim mLayer  
Dim mWorkspace  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
Set mMap = Server.CreateObject( "aims.Map" )  
Set mShapeWorkspace = Server.CreateObject( "aims.ShapeWorkspace" )  
mShapeWorkspace.Name = "Roads"  
mShapeWorkspace.Directory = "D:\EsriData\World"  
mShapeWorkspace.FeatureClass = imsFeatureClass.imsLine  
mMap.InitMap mArcIMSConnector, "World"  
Set mLayer = mMap.Layers.Create( mShapeWorkspace )  
mMap.Layers.Add( mLayer )  
mMap.Refresh  
Response.write mLayer.Workspace.Name
```

### See Also

[ShapeWorkspace](#)  
[SDEWorkspace](#)  
[Layers.Create](#)  
[imsFeatureClass](#)

## **Fields.Count property**

### **Description**

The Fields.Count property returns the number of fields.

### **Syntax**

Fields.Count

### **Arguments**

None

### **Returned Value**

Long            None            Number of fields.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mFields  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set mFields = mMap.Layers.Item(1).Recordset.Fields  
mCount = mFields.Count  
Response.Write CStr(mCount)
```

## Fields.FieldValue method

### Description

The Fields.FieldValue method returns the variant value of the field by index.

### Syntax

Fields.FieldValue(index)

### Arguments

index	Long	None	Index to reference the value.
-------	------	------	-------------------------------

### Returned Value

Variant	None	Value of the field.
---------	------	---------------------

### See Also

[Fields.FieldValueAsString](#)

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim mFields  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set mFields = mMap.Layers.Item(1).Recordset.Fields  
mValue = mFields.FieldValue(1)
```

### Note

Also refer to samples:

Sample10.asp

## **Fields.FieldValueAsString method**

### **Description**

The Fields.FieldValueAsString method returns the string value of the field by index.

### **Syntax**

Fields.FieldValueAsString(index)

### **Arguments**

index	Long	Index to reference the value.
-------	------	-------------------------------

### **Returned Value**

String	None	String value of the field.
--------	------	----------------------------

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mFields  
  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSSMapService"  
  
Set mFields = mMap.Layers.Item(1).Recordset.Fields  
mValue = mFields.FieldValueAsString(1)  
Response.write mValue
```

### **See Also**

[Fields.FieldValue](#)

### **Note**

Also refer to samples:

[Sample11.asp](#)

## **Filter.AddGObject method**

### **Description**

The Filter.AddGObject method adds the object to the spatial filter collection.

### **Syntax**

Filter.AddGObject(GObject)

### **Arguments**

GObject      Object      None      Spatial filter object.

### **Returned Value**

None

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim spobj  
Dim origLayer  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSService"  
Set spobj = Server.CreateObject("aims.Envelope")  
spobj.Xmin = -122.4365  
spobj.Ymin = 37.784  
spobj.Xmax = -122.433  
spobj.Ymax = 37.785  
Set origLayer = mMap.Layers.Item(2).Clone()  
mMap.Layers.Add origLayer  
origLayer.Name = "Filter"  
origLayer.Filter.AddGObject spObj  
mLayers.Add origLayer  
mMap.Refresh
```

## **See Also**

[Filter.RemoveGObject](#)  
[Filter.ClearGObjects](#)  
[Filter.GObjects](#)  
[Filter.Count](#)

## **Note**

Also refer to samples:

[Sample7.asp](#)  
[Sample8.asp](#)  
[Sample11.asp](#)  
[Sample12.asp](#)

## **Filter.Buffer property**

### **Description**

The Filter.Buffer property contains the buffer area width.

### **Syntax**

Filter.Buffer

### **Arguments**

None

### **Returned Value**

Double            0            Buffer area width.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim spobj  
Dim origLayer  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSService"  
Set spobj = Server.CreateObject("aims.Envelope")  
spobj.Xmin = -122.4365  
spobj.Ymin = 37.784  
spobj.Xmax = -122.433  
spobj.Ymax = 37.785  
Set origLayer = mMap.Layers.Item(2).Clone()  
origLayer.Renderer.Symbol.Color = 255  
origLayer.Name = "Filter"  
origLayer.Filter.AddGObject spObj  
origLayer.Filter.BufferUnits = 1  
origLayer.Filter.Buffer = 10  
Set lay = mMap.Layers.CreateBuffer(origLayer.Filter, origLayer)
```

```
mLayers.Add lay  
 mMap.Refresh
```

### **See Also**

[Filter.BufferUnits](#)

### **Note**

Also refer to samples:  
[Sample12.asp](#)

## **Filter.BufferUnits property**

### **Description**

The Filter.BufferUnits property specifies the units that apply to the buffer.

### **Syntax**

Filter.BufferUnits

### **Arguments**

None

### **Returned Value**

Long	0	Specifies units that apply to buffer. Use imsMapUnits constant.
------	---	---

### **Example**

```
Dim mArcIMSConnector
Dim mMap
Dim spobj
Dim origLayer
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSService"
Set spobj = Server.CreateObject("aims.Envelope")
spobj.Xmin = -122.4365
spobj.Ymin = 37.784
spobj.Xmax = -122.433
spobj.Ymax = 37.785
Set origLayer = mMap.Layers.Item(2).Clone()
origLayer.Renderer.Symbol.Color = 255
origLayer.Name = "Filter"
origLayer.Filter.AddGObject spObj
origLayer.Filter.BufferUnits = 1
origLayer.Filter.Buffer = 10
Set lay = mMap.Layers.CreateBuffer(origLayer.Filter, origLayer)
```

```
mLayers.Add lay  
 mMap.Refresh
```

### **See Also**

[Filter.BufferDistance](#)  
[imsMapUnits](#)

### **Note**

Also refer to samples:  
[Sample12.asp](#)

## **Filter.ClearGObjects method**

### **Description**

The Filter.ClearGObjects method removes all objects from the spatial filter collection.

### **Syntax**

Filter.ClearGObjects()

### **Arguments**

None

### **Returned Value**

None

### **Example**

```
Dim mFilter  
Dim mPointObject  
Set mFilter = Server.CreateObject("aims.Filter")  
Set mPointObject = Server.CreateObject("aims.PointObject")  
mFilter.AddGObject( mPointObject )  
mFilter.ClearGObjects()
```

### **See Also**

[Filter.AddGObject](#)

[Filter.RemoveGObject](#)

### **Note**

Also refer to samples:

[Sample7.asp](#)

[Sample11.asp](#)

## **Filter.Count property**

### **Description**

The Filter.Count property returns the number of spatial filter objects.

### **Syntax**

Filter.Count

### **Arguments**

None

### **Returned Value**

Long	None	Number of spatial filter objects.
------	------	-----------------------------------

### **Example**

```
Dim mFilter
Dim mPointObject
Set mFilter = Server.CreateObject("aims.Filter")
Set mPointObject = Server.CreateObject("aims.PointObject")
mPointObject.x = -125.0
mPointObject.y = 45.0

mFilter.AddGObject mPointObject
mCount = mFilter.Count
Response.write CStr(mCount)
```

### **See Also**

[Filter.AddGObject](#)  
[Filter.ClearGObjects](#)  
[Filter.RemoveGObject](#)

## **Filter.GObject method**

### **Description**

The Filter.GObject method returns the object from the spatial filter collection by index.

### **Syntax**

Object Filter.GObject(index)

### **Arguments**

index            Long            Index for reference to the object.

### **Returned Value**

Object            None            Spatial filter object.

### **Example**

```
Dim mFilter  
Dim mPointObject  
Dim mGObject  
Set mFilter = Server.CreateObject("aims.Filter")  
Set mPointObject = Server.CreateObject("aims.PointObject")  
mFilter.AddGObject mPointObject  
Set mGObject = mFilter.GObject(1)
```

### **See Also**

[Filter.AddGObject](#)

[Filter.RemoveGObject](#)

[Filter.ClearGObjects](#)

## **Filter.JoinTables property**

### **Description**

The Filter.JoinTables property contains a list of join tables separated by blank spaces (only for an ArcSDE layer).

### **Syntax**

Filter.JoinTables

### **Arguments**

None

### **Returned Value**

String	None	List of join tables separated by blank spaces.
--------	------	--

### **Example**

```
Dim mFilter  
Set mFilter = Server.CreateObject("aims.Filter")  
mFilter.WhereExpression = "table1.attr1 = table2.attr1"  
mFilter.JoinTables = "Table1 Table2"
```

### **See Also**

[Filter.WhereExpression](#)

## **Filter.RemoveGObject method**

### **Description**

The Filter.RemoveGObject method removes the object from the spatial filter collection by index.

### **Syntax**

Filter.RemoveGObject(index)

### **Arguments**

index	Long	None	Index for reference to object.
-------	------	------	--------------------------------

### **Returned Value**

Boolean	True	True indicates success. False indicates an error.
---------	------	---

### **Example**

```
Dim mFilter
Dim mPointObject
Set mFilter = Server.CreateObject("aims.Filter")
Set mPointObject = Server.CreateObject("aims.PointObject")
mFilter.AddGObject(mPointObject)
mResult = mFilter.RemoveGObject(1)
```

### **See Also**

- Filter.AddGObject
- Filter.ClearGObjects
- Filter.RemoveGObjects
- Filter.Count

## **Filter.WhereExpression property**

### **Description**

The Filter.WhereExpression property contains the where condition.

### **Syntax**

Filter.WhereExpression

### **Arguments**

None

### **Returned Value**

String            Where condition.

### **Example**

```
Dim mFilter  
Set mFilter = Server.CreateObject("aims.Filter")  
mFilter.WhereExpression = "#ID# < 1976"
```

### **Note**

Also refer to samples:

Sample10.asp

## **GradientFillSymbol.Antialiasing property**

### **Description**

The GradientFillSymbol.Antialiasing property returns or sets a Boolean value indicating whether or not antialiasing will be used while rendering. GradientFillSymbol. Antialiasing allows lines to appear smooth by coloring adjacent pixels in such a way as to give the edge of a line a more gradual fade to the background.

### **Syntax**

GradientFillSymbol.Antialiasing

### **Arguments**

None

### **Returned Value**

Boolean      Turns antialiasing on/off. The default value is False.

### **Example**

```
Dim mGradientFillSymbol  
Set mGradientFillSymbol = Server.CreateObject("aims.GradientFillSymbol")  
mGradientFillSymbol.Antialiasing = True
```

## **GradientFillSymbol.Boundary property**

### **Description**

The GradientFillSymbol.Boundary property returns or sets a Boolean value indicating whether or not a boundary will be rendered for a GradientFillSymbol.

### **Syntax**

GradientFillSymbol.Boundary

### **Arguments**

None

### **Returned Value**

Boolean      Determines if a boundary will be drawn. The default value is False.

### **Example**

```
dim mGradientFillSymbol  
set mGradientFillSymbol = Server.CreateObject("aims.GradientFillSymbol")  
mGradientFillSymbol.Boundary = True
```

## **GradientFillSymbol.Clone method**

### **Description**

The GradientFillSymbol.Clone method clones the GradientFillSymbol.

### **Syntax**

GradientFillSymbol.Clone ()

### **Arguments**

None

### **Returned Value**

GradientFillSymbol      Cloned symbol.

### **Example**

```
dim mClone  
dim mGradientFillSymbol  
Set mGradientFillSymbol = Server.CreateObject("aims.GradientFillSymbol")  
Set mClone = mGradientFillSymbol.Clone()
```

## **GradientFillSymbol.FinishColor property**

### **Description**

The GradientFillSymbol.FinishColor property returns or sets the ending color constant for a GradientFillSymbol.

### **Syntax**

GradientFillSymbol.FinishColor

### **Arguments**

None

### **Returned Value**

imsColor                   Ending color constant. The default value is imsGreen (65280).

### **Example**

```
Dim mGradientFillSymbol  
Set mGradientFillSymbol = Server.CreateObject("aims.GradientFillSymbol")  
mGradientFillSymbol.FinishColor = imsColor.imsBlue
```

## **GradientFillSymbol.GradientStyle property**

### **Description**

The GradientFillSymbol.GradientStyle property returns or sets a constant indicating the fill style of the GradientFillSymbol. The possible values are:

- 0 = imsBDiagonal
- 1 = imsFDiagonal
- 2 = imsHorizontal
- 3 = imsVertical

### **Syntax**

GradientFillSymbol.GradientStyle

### **Arguments**

None

### **Returned Value**

imsGradientStyle      Gradient fill style constant. The default value is imsBDiagonal (0).

### **See Also**

imsGradientStyle

### **Example**

```
Dim mGradientFillSymbol  
Set mGradientFillSymbol = Server.CreateObject("aims.GradientFillSymbol")  
mGradientFillSymbol.GradientStyle = imsGradientStyle.imsVertical
```

## **GradientFillSymbol.Overlap property**

### **Description**

The GradientFillSymbol.Overlap property returns or sets a Boolean value indicating whether or not labels will be allowed to overlap a GradientFillSymbol.

### **Syntax**

GradientFillSymbol.Overlap

### **Arguments**

None

### **Returned Value**

Boolean      True indicates that labels will be allowed to overlap the GradientFillSymbol. False indicates that they will not. The default value is True.

### **Example**

```
Dim mGradientFillSymbol  
Set mGradientFillSymbol = Server.CreateObject("aims.GradientFillSymbol")  
mGradientFillSymbol.Overlap = True
```

## **GradientFillSymbol.StartColor property**

### **Description**

The GradientFillSymbol.StartColor returns or sets a color constant for the Start color of a GradientFillSymbol.

### **Syntax**

GradientFillSymbol.StartColor

### **Arguments**

None

### **Returned Value**

imsColor                 Start color. The default value is imsRed (255).

### **Example**

```
Dim mGradientFillSymbol  
Set mGradientFillSymbol = Server.CreateObject("aims.GradientFillSymbol")  
mGradientFillSymbol.StartColor = imsColor.imsRed
```

## **GradientFillSymbol.Transparency property**

### **Description**

The GradientFillSymbol.Transparency property returns or sets the transparency coefficient for a GradientFillSymbol. Smaller numbers indicate more transparency.

### **Syntax**

GradientFillSymbol.Transparency

### **Arguments**

None

### **Returned Value**

Double      Transparency coefficient. The default value is 1.0.

### **Example**

```
Dim mGradientFillSymbol  
Set mGradientFillSymbol = Server.CreateObject("aims.GradientFillSymbol")  
mGradientFillSymbol.Transparency = 1.0
```

## **GroupRenderer.Add method**

### **Description**

The GroupRenderer.Add method adds a renderer to the GroupRenderer object.

### **Syntax**

```
GroupRenderer.Add( Renderer )
```

### **Arguments**

Renderer      Renderer object to add.

### **Returned Value**

Boolean      True indicates renderer was added successfully and False indicates an error.

### **Example**

```
Dim mGroupRenderer  
Dim mRenderer  
Set mGroupRenderer = Server.CreateObject( "aims.GroupRenderer" )  
Set mRenderer = Server.CreateObject( "aims.SimpleRenderer" )  
mResult = mGroupRenderer.Add( mRenderer )
```

## **GroupRenderer.Clear method**

### **Description**

The GroupRenderer.Clear method removes all renderers from the GroupRenderer.

### **Syntax**

GroupRenderer.Clear()

### **Arguments**

None

### **Returned Value**

None

### **Example**

```
Dim mGroupRenderer  
Dim mRenderer  
Set mGroupRenderer = Server.CreateObject("aims.GroupRenderer")  
Set mRenderer = Server.CreateObject("aims.SimpleRenderer")  
mResult = mGroupRenderer.Add( mRenderer )  
mGroupRenderer.Clear()
```

## **GroupRenderer.Clone method**

### **Description**

The GroupRenderer.Clone method clones the GroupRenderer.

### **Syntax**

GroupRenderer GroupRenderer.Clone()

### **Arguments**

None

### **Returned Value**

GroupRenderer            Cloned renderer.

### **Example**

```
Dim mClone  
Dim mGroupRenderer  
Set mGroupRenderer = Server.CreateObject("aims.GroupRenderer")  
Set mClone = mGroupRenderer.Clone()
```

## **GroupRenderer.Count property**

### **Description**

The GroupRenderer.Count property returns the number of renderers in the GroupRenderer.

### **Syntax**

GroupRenderer.Count

### **Arguments**

None

### **Returned Value**

Long            The number of renderers in the GroupRenderer.

### **Example**

```
Dim mGroupRenderer  
Dim mRenderer  
Set mGroupRenderer = Server.CreateObject("aims.GroupRenderer")  
Set mRenderer = Server.CreateObject("aims.SimpleRenderer")  
mResult = mGroupRenderer.Add( mRenderer )  
mCount = mGroupRenderer.Count
```

## **GroupRenderer.Item method**

### **Description**

The GroupRenderer.Item method returns the renderer in the GroupRenderer at the specified index.

### **Syntax**

GroupRenderer.Item(index)

### **Arguments**

index            A long indicating the position of the renderer in the GroupRenderer collection

### **Returned Value**

Object            Any renderer.

### **Example**

```
Dim mGroupRenderer  
Dim mRenderer  
Set mGroupRenderer = Server.CreateObject("aims.GroupRenderer")  
Set mRenderer = Server.CreateObject("aims.SimpleRenderer")  
mResult = mGroupRenderer.Add( mRenderer )  
Set mRenderer = mGroupRenderer.Item(index)
```

### **Note**

Also refer to samples:  
Buffer.asp

## **GroupRenderer.Remove method**

### **Description**

The GroupRenderer.Remove method removes a renderer from a GroupRenderer by index.

### **Syntax**

GroupRenderer.Remove(index)

### **Arguments**

index            Index to reference to the renderer.

### **Returned Value**

Boolean        Indicates the success of removing the renderer.

### **Example**

```
Dim mGroupRenderer  
Dim mRenderer  
Set mGroupRenderer = Server.CreateObject("aims.GroupRenderer")  
Set mRenderer = Server.CreateObject("aims.SimpleRenderer")  
mResult = mGroupRenderer.Add( mSimpleRenderer )  
mResult = mGroupRenderer.Remove(1)
```

## **HashLineSymbol.Antialiasing property**

### **Description**

The HashLineSymbol.Antialiasing property returns or sets a Boolean value indicating whether or not antialiasing will be used while rendering a HashLineSymbol. Antialiasing allows lines to appear smooth by coloring adjacent pixels in such a way as to give the edge of a line a more gradual fade to the background.

### **Syntax**

HashLineSymbol.Antialising

### **Arguments**

None

### **Returned Value**

Boolean      Turns antialiasing on/off. The default value is False.

### **Example**

```
Dim mHashLineSymbol  
Set mHashLineSymbol = Server.CreateObject("aims.HashLineSymbol")  
mHashLineSymbol.Antialising = True
```

## **HashLineSymbol.Clone method**

### **Description**

The HashLineSymbol.Clone method clones the HashLineSymbol.

### **Syntax**

HashLineSymbol.Clone()

### **Arguments**

None

### **Returned Value**

HashLineSymbol      Cloned symbol.

### **Example**

```
Dim mClone  
Dim mHashLineSymbol  
Set mHashLineSymbol = Server.CreateObject("aims.HashLineSymbol")  
Set mClone = mHashLineSymbol.Clone()
```

## **HashLineSymbol.Color property**

### **Description**

The HashLineSymbol.Color property returns or sets a color constant for the HashLineSymbol.

### **Syntax**

HashLineSymbol.Color

### **Arguments**

None

### **Returned Value**

imsColor                  HashLineSymbol color. The default value is imsBlack (0).

### **Example**

```
Dim mHashLineSymbol  
Set mHashLineSymbol = Server.CreateObject("aims.HashLineSymbol")  
mHashLineSymbol.Color = imsColor.imsRed
```

### **See Also**

imsColor

## **HashLineSymbol.Interval property**

### **Description**

The HashLineSymbol.Interval property returns or sets the distance between railroad crosshatches on the HashLineSymbol. The distance units are pixels.

### **Syntax**

HashLineSymbol.Interval

### **Arguments**

None

### **Returned Value**

Long            Distance between railroad crosshatches. The default value is 8.

### **Example**

```
Dim mHashLineSymbol  
Set mHashLineSymbol = Server.CreateObject("aims.HashLineSymbol")  
mHashLineSymbol.Interval = 10
```

## **HashLineSymbol.LineThickness property**

### **Description**

The HashLineSymbol.LineThickness property returns or sets the line thickness in pixels of the HashLineSymbol.

### **Syntax**

HashLineSymbol.LineThickness

### **Arguments**

None

### **Returned Value**

Long            Line thickness. The default value is 1.

### **Example**

```
Dim mHashLineSymbol  
Set mHashLineSymbol = Server.CreateObject("aims.HashLineSymbol")  
mHashLineSymbol.LineThickness = 3
```

## **HashLineSymbol.Overlap property**

### **Description**

The HashLineSymbol.Overlap property returns or sets a Boolean value indicating whether or not labels will overlap a HashLineSymbol.

### **Syntax**

HashLineSymbol.Overlap

### **Arguments**

None

### **Returned Value**

Boolean      True indicates that labels will be allowed to overlap the HashLineSymbol. False indicates that labels will not overlap the symbol. The default value is True.

### **Example**

```
Dim mHashLineSymbol  
Set mHashLineSymbol = Server.CreateObject("aims.HashLineSymbol")  
mHashLineSymbol.Overlap = False
```

## **HashLineSymbol.Style** property

### **Description**

The HashLineSymbol.Style property returns or sets a style constant that specifies a line style for a HashLineSymbol. The possible values are:

- 0 = imsForeground
- 1 = imsBackGround

### **Syntax**

HashLineSymbol.Style

### **Arguments**

None

### **Returned Value**

imsGroundStyle      Line style. The default value is imsForeground (0).

### **Example**

```
Dim mHashLineSymbol  
Set mHashLineSymbol = Server.CreateObject("aims.HashLineSymbol")  
mHashLineSymbol.Style = imsGroundStyle.imsBackGround
```

### **See Also**

imsGroundStyle

## **HashLineSymbol.TickThickness property**

### **Description**

The HashLineSymbol.TickThickness property returns or sets the tick thickness in pixels for the HashLineSymbol.

### **Syntax**

HashLineSymbol.TickThickness

### **Arguments**

None

### **Returned Value**

Long            Tick thickness. The default value is 1.

### **Example**

```
Dim mHashLineSymbol  
Set mHashLineSymbol = Server.CreateObject("aims.HashLineSymbol")  
mHashLineSymbol.TickThickness = 2
```

## **HashLineSymbol.Transparency property**

### **Description**

The HashLineSymbol.Transparency property returns or sets the transparency coefficient for the HashLineSymbol. Lower numbers will display the symbol with greater transparency.

### **Syntax**

HashLineSymbol.Transparency

### **Arguments**

None

### **Returned Value**

Double Transparency coefficient. The default value is 1.0.

### **Example**

```
Dim mHashLineSymbol  
Set mHashLineSymbol = Server.CreateObject("aims.HashLineSymbol")  
mHashLineSymbol.Transparency = 1.0
```

## **HashLineSymbol.Width property**

### **Description**

The HashLineSymbol.Width property returns or sets the symbol width in pixels of a HashLineSymbol.

### **Syntax**

HashLineSymbol.Width

### **Arguments**

None

### **Returned Value**

Double      Symbol width. The default value is 6.

### **Example**

```
Dim mHashLineSymbol  
Set mHashLineSymbol = Server.CreateObject("aims.HashLineSymbol")  
mHashLineSymbol.Width = 10
```

## **ImageLayer.Id property**

### **Description**

The ImageLayer.Id property returns or sets a string identifier for the image layer.

### **Syntax**

ImageLayer.Id

### **Arguments**

None

### **Returned Value**

String            None            Id of the image layer.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
mId = mMap.Layers.Item(1).Id  
Response.write mId
```

## **ImageLayer.MaxScale property**

### **Description**

The ImageLayer.MaxScale property contains the maximum scale to display the layer.

### **Syntax**

`ImageLayer.MaxScale`

### **Arguments**

None

### **Returned Value**

`Double            0            Maximum scale value.`

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSSMapService"  
mMap.Layers.Item(1).MaxScale = 25000  
mMap.Refresh
```

### **See Also**

[ImageLayer.MinScale](#)

## **ImageLayer.MinScale property**

### **Description**

The ImageLayer.MinScale property contains the minimum scale to display the layer.

### **Syntax**

ImageLayer.MinScale

### **Arguments**

None

### **Returned Value**

Double            0            Minimum scale value.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
mMap.Layers.Item(1).MinScale = 2500  
mMap.Refresh
```

### **See Also**

[ImageLayer.MaxScale](#)

## **ImageLayer.Name property**

### **Description**

The ImageLayer.Name property returns or sets the name of the image layer.

### **Syntax**

ImageLayer.Name

### **Arguments**

None

### **Returned Value**

String            None            Name of the image layer.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSSMapService"  
 mName = mMap.Layers.Item(1).Name  
Response.write mName
```

## **ImageLayer.Visible property**

### **Description**

The ImageLayer.Visible property sets the visibility of the layer.

### **Syntax**

ImageLayer.Visible

### **Arguments**

None

### **Returned Value**

Boolean      True      True indicates layer is visible. False indicates layer is invisible.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
mMap.Layers.Item(1).Visible = True  
mMap.Refresh
```

## **ImageLayer.Workspace property**

### **Description**

The ImageLayer.Workspace property exists only for the layer created through method Layers.Create(). It contains the path to the data.

### **Syntax**

ImageLayer.Workspace

### **Arguments**

None

### **Returned Value**

ImageWorkspace	None	Workspace object for image layer.
----------------	------	-----------------------------------

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mImageWorkspace  
Dim mLayer  
Dim mWorkspace  
  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
  
Set mMap = Server.CreateObject( "aims.Map" )  
Set mImageWorkspace = Server.CreateObject( "aims.ImageWorkspace" )  
  
mImageWorkspace.Name = "World.tif"  
mImageWorkspace.Directory = "D:\EsriData\World"  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set mLayer = mMap.Layers.Create(mImageWorkspace)  
mMap.Layers.Add mLayer  
Set mWorkspace = mLayer.Workspace  
mMap.Refresh
```

### **See Also**

[Layers.Create](#)  
[ImageWorkspace](#)

## **ImageWorkspace.Directory property**

### **Description**

The ImageWorkspace.Directory property contains a directory containing images.

### **Syntax**

ImageWorkspace.Directory

### **Arguments**

None

### **Returned Value**

String            None            Directory containing images.

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mImageWorkspace
```

```
Dim mLayer
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
```

```
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
```

```
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mImageWorkspace = Server.CreateObject("aims.ImageWorkspace")
```

```
mImageWorkspace.Directory = "C:\Esridata\Images"
```

```
mImageWorkspace.Name = "World.tif"
```

```
Set mLayer = MMap.Create(mImageWorkspace)
```

```
mMap.Layers.Add mLayer
```

```
mMap.Refresh
```

## **ImageWorkspace.Id property**

### **Description**

The ImageWorkspace.Id property returns or sets a string identifier for the Image Workspace.

### **Syntax**

ImageWorkspace.Id

### **Arguments**

None

### **Returned Value**

String            None            Workspace Id.

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mLayer
```

```
Dim mImageWorkspace
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
```

```
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
```

```
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mImageWorkspace = Server.CreateObject("aims.ImageWorkspace")
```

```
mImageWorkspace.Id = "Image Workspace:1"
```

```
mImageWorkspace.Directory = "C:/world/tiffs"
```

```
mImageWorkspace.Name = "world.tiff"
```

```
Set mLayer = mMap.Layers.Create(mImageWorkspace)
```

```
mMap.Layers.Add mLayer
```

```
mMap.Refresh
```

## **ImageWorkspace.Name property**

### **Description**

The ImageWorkspace.Name property returns or sets the name of the file.

### **Syntax**

ImageWorkspace.Name

### **Arguments**

None

### **Returned Value**

String            None            Name of the file.

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mLayer
```

```
Dim mImageWorkspace
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
```

```
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
```

```
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mImageWorkspace = Server.CreateObject("aims.ImageWorkspace")
```

```
mImageWorkspace.Id = "ImageWorkspace:1"
```

```
mImageWorkspace.Directory = "C:/world/tiffs"
```

```
mImageWorkspace.Name = "world.tiff"
```

```
Set mLayer = mMap.Layers.Create(mImageWorkspace)
```

```
mMap.Layers.Add mLayer
```

```
mMap.Refresh
```

## imsArrowType constants

### Description

imsArrowType constants define the type of north arrow.

### Constants

imsType1	1	Type1.	
imsType2	2	Type2.	
imsType3	3	Type3.	
imsType4	4	Type4.	
imsType5	5	Type5.	
imsType6	6	Type6.	
imsType7	7	Type7.	
imsType8	8	Type8.	

### Returned Value

Long            imsType1            Type of north arrow.

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim al  
Dim arrow  
  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
mMap.width = 500  
mMap.Height = 450  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.visible = true  
Set arrow = Server.CreateObject("aims.NorthArrowObject")  
arrow.x = 25  
arrow.y = 25  
arrow.arrowtype = 1  
arrow.size = 25  
al.add arrow, 0  
mMap.Layers.Add al  
mMap.Refresh
```

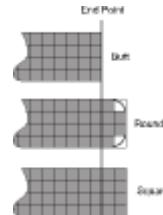
## imsCapStyle constants

### Description

imsCapStyle constants contain line end or cap style.

### Constants

imsRound	0	Round type.
imsButt	1	Butt type.
imsSquare	2	Square type.



### Returned Value

Long              imsRound              Line and style.

### Example

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
mMap.Width = 500
mMap.Height = 400
mMap.Layers.Item(1).Renderer.Symbol.CapStyle = 1
mMap.Refresh
```

## imsColor constants

### Description

imsColor constants contain Color constants.

### Constants

imsBlack	0	Black color constant.
imsRed	255	Red color constant.
imsGreen	65280	Green color constant.
imsBlue	16711680	Blue color constant.
imsMagenta	16711935	Magenta color constant.
imsCyan	16776960	Cyan color constant.
imsWhite	16777215	White color constant.
imsLightgray	13882323	Light gray color constant.
imsDarkgray	11119017	Dark gray color constant.
imsGray	8421504	Gray color constant.
imsLightyellow	14745599	Light yellow color constant.
imsYellow	65535	Yellow color constant.
imsLimegreen	3329330	Lime green color constant.
imsTeal	8421376	Teal color constant.
imsDarkgreen	25600	Dark green color constant.
imsMaroon	128	Maroon color constant.
imsPurple	8388736	Purple color constant.
imsOrange	42495	Orange color constant.
imsKhaki	9234160	Khaki color constant.
imsOlive	32896	Olive color constant.
imsBrown	2763429	Brown color constant.
imsNavy	8388608	Navy color constant.

### Returned Value

Long            None            Color constants.

### Example

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
mMap.Width = 500  
mMap.Height = 400  
mMap.Layers.Item(1).Renderer.Symbol.Color = 255  
mMap.Refresh
```

## imsDirection constants

### Description

imsDirection constants define the pan direction.

### Constants

imsNone	0	No direction.
imsNorth	1	North direction.
imsNorthEast	2	Northeast direction.
imsEast	3	East direction.
imsSouthEast	4	Southeast direction.
imsSouth	5	South direction.
imsSouthWest	6	Southwest direction.
imsWest	7	West direction.
imsNorthWest	8	Northwest direction.

### Returned Value

Long            None            Defines direction.

### Example

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
mMap.Width = 500
mMap.Height = 400
mMap.Refresh
```

```
If Action = "NorthEast" Then
    mMap.DoPan 2, 3
    mMap.Refresh
end if
```

## **imsFeatureClass constants**

### **Description**

imsFeatureClass constants contain the type of featureclass layer.

### **Constants**

imsPoint	1	Point.
imsLine	2	Line.
imsPolygon	3	Polygon.

### **Returned Value**

Long              None              Featureclass layer type.

### **Example**

imsFeatureClass.imsLine

## imsFillStyle constants

### Description

imsFillStyle constants contain the fill type.

### Constants

imsSolid	0	Solid fill type
imsBDiagonal	1	BDiagonal fill type.
imsFDiagonal	2	FDiagonal fill type.
imsCross	3	Cross fill type.
imsDiagcross	4	Diagcross fill type.
imsHorizontal	5	Horizontal fill type.
imsVertical	6	Vertical fill type.
imsLightgray	7	Light gray fill type.
imsGray	8	Gray fill type.
imsDarkgray	9	Dark gray fill type.

### Returned Value

Long            imsSolid            Fill type.

### Example

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
mMap.Width = 500
mMap.Height = 400
mMap.Layers.Item(1).Renderer.Item(1).Symbol.FillStyle = 1
mMap.Refresh
```

## imsFontStyle constants

### Description

imsFontStyle constants contain the font style.

### Constants

imsRegular	0	Regular font style.
imsBold	1	Bold font style.
imsItalic	2	Italic font style.
imsUnderline	3	Underline font style.
imsOutline	4	Outline font style.

### Returned Value

Long              imsRegular              Font style.

### Example

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSSMapService"
mMap.Width = 500
mMap.Height = 400
mMap.Layers.Item(1).Renderer.Item(2).Symbol.FontStyle = 1
mMap.Refresh
```

## imsGlobalColor constants

### Description

imsGlobalColor constants contain color constants.

### Constants

imsAliceblue	16775408	Aliceblue color constant.
imsAntiquewhite	14150650	Antique white color constant.
imsAqua	16776960	Aqua color constant.
imsAquamarine	13959039	Aquamarine color constant.
imsAzure	16777200	Azure color constant.
imsBeige	14880885	Beige color constant.
imsBisque	12903679	Bisque color constant.
imsBlack	0	Black color constant.
imsBlanchedalmond	13495295	Blanched almond color constant.
imsBlue	16711680	Blue color constant
imsBlueViolet	14822282	Blue violet color constant
imsBrown	2763429	Brown color constant
imsCadetblue	10526303	Cadet blue constant.
imsChartreuse	65407	Chartreuse color constant.
imsChocolate	1993170	Chocolate color constant.
imsCoral	5275647	Coral color constant.
imsCornflowerblue	15570276	Cornflower blue color constant.
imsCornsilk	14481663	Cornsilk color constant.
imsCrimson	3937500	Crimson color constant.
imsCyan	16776960	Cyan color constant.
imsDarkblue	9109504	Dark blue color constant.
imsDarkcyan	9145088	Dark cyan color constant.
imsDarkgoldenrod	755384	Dark goldenrod color constant.
imsDarkgray	11119017	Dark gray color constant.
imsDarkgreen	25600	Dark green color constant.
imsDarkkhaki	7059389	Dark khaki color constant.
imsDarkmagenta	9109643	Dark magenta color constant.
imsDarkolivegreen	2845525	Dark olive green color constant.
imsDarkorange	36095	Dark orange color constant.

imsDarkred	139	Dark red color constant.
imsDarksalmon	8034025	Dark salmon color constant.
imsDarkseagreen	9419919	Dark sea green color constant.
imsDarkslateblue	9125192	Dark slate blue color constant.
imsDarkslategray	5197615	Dark slate gray color constant.
imsDarkturquoise	13749760	Dark turquoise color constant.
imsDarkviolet	13828244	Dark violet color constant.
imsDeppink	9633812	Deep pink color constant.
imsDeepskyblue	16760576	Deep sky blue color constant.
imsDimgray	6908265	Dim gray color constant.
imsDodgerblue	16748574	Dodger blue color constant.
imsFirebrick	2237106	Firebrick color constant.
imsFloralwhite	15792895	Floral white color constant.
imsForestgreen	2263842	Forest green color constant.
imsFuchsia	16711935	Fuchsia color constant.
imsGainsboro	14474460	Gainsboro color constant.
imsGhostwhite	16775416	Ghost white color constant.
imsGold	55295	Gold color constant.
imsGoldenrod	2139610	Goldenrod color constant.
imsGray	8421504	Gray color constant.
imsGreen	32768	Green color constant.
imsGreenyellow	3145645	Green-yellow color constant.
imsHoneydey	15794160	Honeydey color constant.
imsHotPink	11823615	Hot pink color constant.
imsIndianred	6053069	Indian red color constant.
imsIndigo	8519755	Indigo color constant.
imsIvory	15794176	Ivory color constant.
imsKhaki	9234160	Khaki color constant.
imsLavender	15394534	Lavender color constant.
imsLavenderblush	16117775	Lavender blush color constant
imsLawngreen	64636	Lawn green color constant.
imsLemonchiffon	13499135	Lemon chiffon color constant.

imsLightblue	15128749	Light blue color constant
imsLightcoral	8421616	Light coral color constant
imsLightcyan	16777184	Light cyan color constant.
imsLightgoldenrodyellow	13826810	Light goldenrod yellow color constant
imsLightGreen	9498256	Light green color constant
imsLightgray	13882323	Light gray color constant
imsLightpink	12695295	Light pink color constant.
imsLightsalmon	7446783	Light salmon color constant
imsLightseagreen	11186720	Light sea green color constant
imsLightskyblue	15388295	Light sky blue color constant
imsLightslategray	10061943	Light slate gray color constant
imsLightsteelblue	14599344	Light steel blue color constant
imsLightyellow	14745599	Light yellow color constant
imsLime	65280	Lime color constant.
imsLimegreen	332933	Lime green color constant.
imsLinen	15134970	Linen color constant.
imsMagenta	16711935	Magenta color constant.
imsMaroon	128	Maroon color constant.
imsMediumaquamarine	11193702	Medium aquamarine color constant
imsMediumblue	13434880	Medium blue color constant
imsMediumorchid	13850042	Medium orchid color constant
imsMediumpurple	14381203	Medium purple color constant.
imsMediumseagreen	7451452	Medium sea green color constant.
imsMediumslateblue	15624315	Medium slate blue color constant.
imsMediumspringgreen	10156544	Medium spring green color constant.
imsMediumturquoise	13422920	Medium turquoise color constant.
imsMediumvioletred	8721863	Medium violet red color constant.
imsMidnightblue	7346457	Midnight blue color constant.
imsMintcream	16449525	Mint cream color constant.
imsMistyrose	14804223	Misty rose color constant.
imsMoccasin	11920639	Moccasin color constant.
imsNavajowhite	11394815	Navajo white color constant.

imsNavy	8388608	Navy color constant.
imsOldlace	14087677	Old lace color constant.
imsOlive	32896	Olive color constant.
imsOlivedrab	2330219	Olive drab color constant.
imsOrange	42495	Orange color constant.
imsOrangered	17919	Orange red color constant.
imsOrchid	14053594	Orchid color constant.
imsPalegoldenrod	11200750	Pale goldenrod color constant.
imsPalegreen	10021784	Pale green color constant.
imsPaleturquoise	15658671	Pale turquoise color constant.
imsPalevioletred	9662683	Pale violet red color constant.
imsPapayawhip	14018047	Papaya whip color constant.
imsPeachpuff	12180223	Peach puff color constant.
imsPeru	4163021	Peru color constant.
imsPink	13353215	Pink color constant.
imsPowderblue	15130800	Powder blue color constant.
imsPurple	8388736	Purple color constant
imsRed	255	Red color constant
imsRosybrown	9408444	Rosy brown color constant
imsRoyalblue	14772545	Royal blue color constant.
imsSaddlebrown	1262987	Saddle brown color constant.
imsSalmon	7504122	Salmon color constant.
imsSandybrown	6333690	Sandy brown color constant.
imsSeagreen	5737262	Sea green color constant.
imsSeashell	15660543	Seashell color constant.
imsSienna	2970272	Sienna color constant.
imsSilver	12632256	Silver color constant.
imsSkyblue	15453831	Sky blue color constant.
imsSlateblue	13458026	Slate blue color constant.
imsSlategray	9470064	Slate gray color constant.
imsSnow	16448255	Snow color constant.
imsSteelblue	11829830	Steel blue color constant.

imsTan	9221330	Tan color constant.
imsThistle	14204888	Thistle color constant.
imsTomato	4678655	Tomato color constant.
imsTurquoise	13688896	Turquoise color constant.
imsViolet	15631086	Violet color constant.
imsWheat	11788021	Wheat color constant.
imsWhite	16777215	White color constant.
imsWhitesmoke	16119285	White smoke color constant.
imsYellow	65535	Yellow color constant.
imsYellowgreen	3329434	Yellow-green color constant.

#### **Returned Value**

Long            None            Color constants.

#### **Example**

`imsGlobalColor.imsMediumaquamarine`

## **imsGradientStyle constants**

### **Description**

imsGradientStyle constants contain the gradient type.

### **Constants**

imsBDiagonal	0	BDiagonal gradient type.
imsFDiagonal	1	FDiagonal gradient type.
imsHorizontal	2	Horizontal gradient type.
imsVertical	3	Vertical gradient type.

### **Returned Value**

Long              imsBDiagonal              Gradient type.

### **Example**

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
mMap.Width = 500
mMap.Height = 400
mMap.Layers.Item(1).Renderer.Item(1).Symbol.GradientStyle = 1
mMap.Refresh
```

## **imsGroundStyle constants**

### **Description**

imsGroundStyle constants contain the symbol type.

### **Constants**

imsForeGround	0	Foreground type.
imsBackGround	1	Background type.

### **Returned Value**

Long              imsForeGround              Symbol type

### **Example**

imsGroundStyle.imsForeGround

## imsHAlignment constants

### Description

imsHAlignment constants contain the horizontal alignment.

### Constants

imsLeft	0	Left alignment
imsCenter	1	Center alignment
imsRight	2	Right alignment

### Returned Value

Long              imsRight              Horizontal alignment.

### Example

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSSMapService"
mMap.Width = 500
mMap.Height = 400
Set al = Server.CreateObject("aims.AcetateLayer")
Al.visible = true
Set textobj = Server.CreateObject("aims.TextObject")
textobj.x = 25
textobj.y = 25
textobj.Label = "This is the Text!"
textobj.TextMarkerSymbol.Halignment = 1
al.Add textobj, 0
mMap.Layers.Add al
mMap.Refresh
```

## **imsHMLabels constants**

### **Description**

imsHMLabels constants determine how many labels are to be drawn to label a feature.

### **Constants**

imsOne_label_per_name	0	Labels one label per feature name.
imsOne_label_per_shape	1	Labels one label per feature even if features with the same name exist.
imsOne_label_per_part	2	Labels all parts of a feature (in the case of multi-part features).

### **Returned Value**

Long              None              Determines how many labels are to be drawn to label a feature.

### **Example**

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
mMap.Width = 500
mMap.Height = 400
mMap.Layers.Item(1).Renderer.Item(2).HMLabels = 1
mMap.Refresh
```

## imsJoinStyle constants

### Description

imsJoinStyle constants contain the line intersect join types.

### Constants

imsRound	0	Round join type.
imsMiter	1	Miter join type.
imsBevel	2	Bevel join type.



### Returned Value

Long              imsRound              Join type.

### Example

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
mMap.Width = 500
mMap.Height = 400
mMap.Layers.Item(1).Renderer.Symbol.JoinStyle = 2
mMap.Refresh
```

## **imsLabelMode constants**

### **Description**

imsLabelMode constants contain the label mode for shield symbols.

### **Constants**

imsFull	0	Takes the full label in the label field and puts it into the shield.
imsNumericOnly	1	Put in the numbers from that label.

### **Returned Value**

Long            imsFull            Label mode.

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
```

```
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
```

```
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
mMap.Width = 500
```

```
mMap.Height = 400
```

```
mMap.Layers.Item(1).Renderer.Symbol.LabelMode = 1
```

```
mMap.Refresh
```

## imsLineStyle constants

### Description

imsLineStyle constants contain the line types.

### Constants

imsSolid	0	Solid line type.
imsDash	1	Dash line type.
imsDot	2	Dot line type.
imsDash_dot	3	Dash-dot line type.
imsDash_dot_dot	4	Dash-dot-dot line type.

### Returned Value

Long              imsSolid              Line type.

### Example

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
```

```
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
```

```
mMap.InitMap mArcIMSConnector, "IMSSMapService"
```

```
mMap.Width = 500
```

```
mMap.Height = 400
```

```
mMap.Layers.Item(1).Renderer.Symbol.Style = 1
```

```
mMap.Refresh
```

## imsLLPosition constants

### Description

imsLLPosition constants determine where on the line to place the label.

### Constants

imsPlaceNone	0	Do not place a label.
imsPlaceAbove	1	Place above the line.
imsPlaceBelow	2	Place below the line.
imsPlaceOnTop	3	Place on the line.
imsPlaceLeft	4	Place to the left of the line.
imsPlaceRight	5	Place to the right of the line.
imsPlaceAboveBelow	6	Place above or below the line.
imsPlaceLeftRight	7	Place at either end of the line.
imsPlaceInLine	8	Place anywhere on the line.
imsPlaceAtStart	9	Place at the beginning of the line.
imsPlaceAtEnd	10	Place at the end of the line.
imsPlaceAtEitherEnd	11	Place at the beginning or end of the line.
imsPlaceParallel	12	Place parallel to the line.
imsPlacePerpendicular	13	Place perpendicular to the line.
imsPlaceHorizontal	14	Place label so that it is always horizontal.
imsPlaceOnTopHorizontal	15	Place label on top of the line but always horizontal.

### Returned Value

Long            None            Line label position

### Example

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSSMapService"
mMap.Width = 500
mMap.Height = 400
mMap.Layers.Item(1).Renderer.Item(2).LLPosition = 12
mMap.Refresh
```

## imsMapUnits constants

### Description

imsMapUnits constants contain the map unit constants.

### Constants

imsDecimalDegrees	0	Decimal degrees units.
imsMiles	1	Miles units.
imsFeets	2	Feets units.
imsKilometers	3	Kilometers units.
imsMeters	4	Meters units.
imsInches	5	Inches units.
imsCentimeters	6	Centimeters units.

### Returned Value

Long              imsDecimalDegrees              Map units.

### Example

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSSMapService"
mMap.Width = 500
mMap.Height = 400
mMap.MapUnits = 5
mMap.Refresh
```

## imsMarkerType constants

### Description

imsMarkerType constants contain point featureclass layer marker types.

### Constants

imsCircle	0	Circle marker type.
imsTriangle	1	Triangle marker type.
imsSquare	2	Square marker type.
imsCross	3	Cross marker type.
imsStar	4	Star marker type.

### Returned Value

Long              imsCircle              Marker type.

### Example

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSSMapService"
mMap.Width = 500
mMap.Height = 400
mMap.Layers.Item(1).Renderer.Symbol.MarkerType = 3
mMap.Refresh
```

## imsPrintMode constants

### Description

imsPrintMode constants contain label characters printing mode types.

### Constants

imsNone	0	No change is made.
imsTitleCaps	1	The first letter of each word in a label is uppercase and everything else is lowercase.
imsAllUpper	2	All letters are upper case.
imsAllLower	3	All letters are lower case.

### Returned Value

Long            imsNone            Printing mode.

### Example

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
mMap.Width = 500
mMap.Height = 400
Set al = Server.CreateObject("aims.AcetateLayer")
al.visible = true
Set textobj = Server.CreateObject("aims.TextObject")
textobj.x = 25
textobj.y = 25
textobj.Label = "This is the label!"
textobj.TextMarkerSymbol.Printmode = 3
al.add textobj, 0
mMap.Layers.Add al
mMap.Refresh
```

## imsRange constants

### Description

imsRange constants define the type of the symbol for ValueMapLabel and ValueMap renderers.

### Constants

imsExact	0	Exact symbol constant.
imsRange	1	Range symbol constant.
imsOther	2	Other symbol constant.

### Returned Value

Long            None            Type of the symbol.

### Example

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSSMapService"
mMap.Width = 500
mMap.Height = 400
mMap.Layers.Item(1).Renderer.Item(1).Range(2) = 1
mMap.Refresh
```

## imsShieldType constants

### Description

imsShieldType constants contain shield renderer symbol types.

### Constants

imsInterstate	0	Interstate symbol type.
imsUSRoad	1	Usroad symbol type.
imsRect	2	Rect symbol type.
imsOval	3	Oval symbol type.
imsMexican	4	Mexican symbol type.

### Returned Value

Long            None            Symbol type.

### Example

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSSMapService"
mMap.Width = 500
mMap.Height = 400
mMap.Layers.Item(1).Renderer.Symbol.ShieldType = 2
mMap.Refresh
```

## imsUnit constants

### Description

imsUnit constants define units for acetate objects.

### Constants

imsPixel	0	Pixel units.
imsDatabase	1	Database units.

### Returned Value

Long              ims        Pixel units for acetate objects.

### Example

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
mMap.Width = 500
mMap.Height = 400
Set al = Server.CreateObject("aims.AcetateLayer")
al.visible = true
Set pt = Server.CreateObject("aims.PointObject")
pt.x = -117.00
pt.y = 45.00
al.add pt, 1
mMap.Layers.Add al
mMap.Refresh
```

## imsVAlignment constants

### Description

imsVAlignment constants contain vertical alignment of labels or text.

### Constants

imsTop	0	Top alignment.
imsCenter	1	Center alignment.
imsBottom	2	Bottom alignment.

### Returned Value

Long              imsTop              Vertical alignment.

### Example

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSService"
mMap.Width = 500
mMap.Height = 400
Set al = Server.CreateObject("aims.AcetateLayer")
al.visible = true
Set textobj = Server.CreateObject("aims.TextObject")
textobj.x = 25
textobj.y = 25
textobj.label = "This is the Label!"
textobj.TextMarkerSymbol.Valignment = 2
al.add textobj, 0
mMap.Layers.Add al
mMap.Refresh
```

## **imsWeight constants**

### **Description**

imsWeight constants prioritize the importance of labels.

### **Constants**

imsNo_weight	0	No weight constant.
imsMed_weight	1	Medium weight constant.
imsHigh_weight	2	High height constant.

### **Returned Value**

Long                imsHigh\_weight importance of labels.

### **Example**

imsWeight.imsHigh\_weight

## Layers.Add method

### Description

The Layers.Add method adds a layer to the end of the layers collection.

### Syntax

Layers.Add(layer)

### Arguments

layer	Object	None	Layer to add to the layers collection.
-------	--------	------	--

### Returned Value

Boolean	True indicates success. False indicates an error.
---------	---

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim mLayer  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSSMapService"  
Set mLayer = mMap.Layers.Item(1).Clone()  
mMap.Layers.Add mLayer  
mMap.Refresh
```

### Note

Also refer to samples:

- Sample7.asp
- Sample8.asp
- Sample9.asp
- Sample12.asp

## **Layers.Clear method**

### **Description**

The Layers.Clear method clears the layers collection.

### **Syntax**

Layers.Clear()

### **Arguments**

None

### **Returned Value**

None

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
mMap.Layers.Clear()  
mMap.Refresh
```

## **Layers.Count property**

### **Description**

The Layers.Count property returns the number of layers in the layers collection.

### **Syntax**

Layers.Count

### **Arguments**

None

### **Returned Value**

Long            None            Number of layers in the layers collection.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSSMapService"  
mCount = mMap.Layers.Count  
Response.write CStr(mCount)
```

### **Note**

Also refer to samples:

Sample1.asp

## **Layers.Create method**

### **Description**

The Layers.Create method creates a new layer. It can be a feature layer or an image layer. In order to create a feature layer as a Workspace, the parameter passes SDEWorkspace or ShapeWorkspace; in order to create image layer, it passes ImageWorkspace. For the FeatureLayer Recordset, the property will be Nothing.

### **Syntax**

```
Layers.Create(Workspace)
```

### **Arguments**

Workspace	Object	None	Workspace object.
-----------	--------	------	-------------------

### **Returned Value**

Object	None	Just created layer. It can be a FeatureLayer or ImageLayer according to the type of workspace passed as the parameter.
--------	------	--

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mLayer  
Dim mShapeWorkspace
```

```
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mShapeWorkspace = Server.CreateObject( "aims.ShapeWorkspace" )  
mShapeWorkspace.Name = "Country"  
mShapeWorkspace.Directory = "D:\EsriData\World"  
mShapeWorkspace.FeatureClass = imsFeatureClass.imsPolygon
```

```
Set mLayer = mMap.Layers.Create( mShapeWorkspace )  
mMap.Layers.Add mLayer  
mMap.Refresh
```

## **Layers.CreateBuffer method**

### **Description**

The Layers.CreateBuffer method creates a new featureclass layer which contains a constructed buffer zone. TargetLayer will contain only objects that are located in the buffer zone.

### **Syntax**

Layers.CreateBuffer(Filter, Targetlayer)

### **Arguments**

Filter	Filter	None	Base filter, buffer zone will be constructed near objects which are located in the filter.
TargetLayer	FeatureClass	None	Optional parameter. TargetLayer will contain only objects which are located in the buffer zone.

### **Returned Value**

FeatureLayer    None              New featureclass layer which contains constructed buffer zone.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mLayer  
Dim mPointObject  
  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
  
Set mPointObject = Server.CreateObject( "aims.PointObject" )  
mPointObject.X = -122.7  
mPointObject.Y = -37.21  
mMap.Layers.Item(1).Filter.AddGObject( mPointObject )  
mMap.Layers.Item(1).Filter.BufferDistance = 0.5  
  
Set mLayer = mMap.Layers.CreateBuffer( mMap.Layers.Item(1).Filter )  
mMap.Layers.Add mLayer  
  
mMap.Refresh
```

### **Note**

Also refer to samples:

Sample12.asp

## **Layers.CreateHighlight method**

### **Description**

The Layers.CreateHighlight method creates new featureclass layer which contains specified objects only. Recordset of the new layer will contain only highlighted objects. By default a renderer is specified (yellow, 50% transparent).

### **Syntax**

```
FeatureLayer Layers.CreateHighlight(ActiveLayer, FieldName, Values)
```

### **Arguments**

ActiveLayer	FeatureLayer	None	Active layer.
FieldName	String	None	Name of the field.
Values	String	None	Values to highlight delimited by space.

### **Returned Value**

```
FeatureLayer None New featureclass layer.
```

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mLayer
```

```
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject( "aims.Map" )
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mLayer = mMap.Layers.CreateHighlight( mMap.Layers.Item(1), "CNTRY_NAME", "USA",
'RUSSIA' )
```

```
mMap.Layers.Add mLayer
```

## Layers.Insert method

### Description

The Layers.Insert method inserts the layer at the specified position.

### Syntax

Layers.Insert(layer, index)

### Arguments

layer	Object	None	Layer
index	Long	None	Position to insert

### Returned Value

Boolean      True indicates success. False indicates index out of bound.

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim mLayer  
  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
  
Set mLayer = mMap.Layers.Item(1).Clone()  
mMap.Layers.Insert mLayer, 3  
mMap.Refresh
```

## Layers.Item method

### Description

The Layers.Item method returns the layer by index.

### Syntax

Layers.Item(index)

### Arguments

index	Long	None	Description
-------	------	------	-------------

### Returned Value

String      Returns string with buffering instructions to be inserted into ArcXML map image request.

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim mLayer
```

```
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mLayer = mMap.Layers.Item(1)  
Response.write mLayer.Name
```

### Note

Also refer to samples:

Sample1.asp

## **Layers.MoveTo method**

### **Description**

The Layers.MoveTo method moves the layer.

### **Syntax**

Layers.MoveTo(fromIndex, toIndex)

### **Arguments**

fromIndex	Long	None	Start position
toIndex	Long	None	End position

### **Returned Value**

Boolean Description.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
  
mMap.Layers.MoveTo 2, 4  
mMap.Refresh
```

### **Note**

Also refer to samples:  
    Sample4.asp

## **Layers.MoveToBottom method**

### **Description**

The Layers.MoveToBottom method moves the layer to the beginning of the layers collection and the places the layer under other layers.

### **Syntax**

Layers.MoveToBottom(index)

### **Arguments**

index            Long     None    Index to reference to the layer.

### **Returned Value**

Boolean        True indicates success. False indicates index out of bounds.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap
```

```
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
mMap.Layers.MoveToBottom 2  
mMap.Refresh
```

### **See Also**

[Layers.MoveToTop](#)

## **Layers.MoveToTop method**

### **Description**

The Layers.MoveToTop method moves the layer to the end of the layers collection and displays the layer on top of layers.

### **Syntax**

```
Layers.MoveToTop(index)
```

### **Arguments**

index	Long	None	Index to reference to the layer.
-------	------	------	----------------------------------

### **Returned Value**

Boolean	True indicates success. False indicates index out of bounds.
---------	--

### **Example**

```
Dim mArcIMSConnector  
Dim mMap
```

```
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.Init mArcIMSConnector, "IMSMAPService"
```

```
mMap.Layers.MoveToTop 2
```

```
mMap.Refresh
```

### **See Also**

[Layers.MoveToBottom](#)

## **Layers.Remove method**

### **Description**

The Layers.Remove method removes the layer from the layers collection by index.

### **Syntax**

Layers.Remove(index)

### **Arguments**

index            Long     None    Index to reference to layer.

### **Returned Value**

Boolean        True indicates success. False indicates index out of bounds.

### **Example**

Dim mArcIMSConnector

Dim mMap

Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )

mArcIMSConnector.ServerName = "IMSServer"

mArcIMSConnector.ServerPort = 5300

Set mMap = Server.CreateObject( "aims.Map" )

mMap.InitMap mArcIMSConnector, "IMSMAPService"

mMap.Layers.Remove 1

mMap.Refresh

### **Note**

Also refer to samples:

Sample7.asp

Sample9.asp

Sample12.asp

## **Legend.Autoextent property**

### **Description**

The Legend.Autoextent property, if true, automatically extends the legend vertically past the size specified in height, if needed.

### **Syntax**

Legend.Autoextent

### **Arguments**

None

### **Returned Value**

Boolean      Autoextent trigger.

### **Example**

```
Dim mLegend  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.Autoextent = True  
  
mURL = mLegend.URL  
Response.write mURL
```

## **Legend.Background property**

### **Description**

The Legend.Background property contains the legend's background color.

### **Syntax**

Legend.Background

### **Arguments**

None

### **Returned Value**

imsColor                  None                  Legend's background color.

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.Background = 255  
mURL = mLegend.URL  
Response.write mURL
```

### **Note**

Also refer to samples:

    Sample2.asp

## **Legend.CanSplit property**

### **Description**

The Legend.CanSplit property permits the splitting of valuemap layers.

### **Syntax**

Legend.CanSplit

### **Arguments**

None

### **Returned Value**

Boolean      Allows splitting of valuemap layers.

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.CanSplit = True  
mURL = mLegend.URL  
Response.write mURL
```

## **Legend.Cellsspacing property**

### **Description**

The Legend.Cellsspacing property defines the number of pixels to pad between entries.

### **Syntax**

Legend.Cellsspacing

### **Arguments**

None

### **Returned Value**

Long              None              Number of pixels.

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.Cellsspacing = 10  
mURL = mLegend.URL  
Response.write mURL
```

### **Note**

Also refer to samples:

  Sample2.asp

## **Legend.Columns property**

### **Description**

The Legend.Columns property defines the number of columns the splits will apply to in the valuemap.

### **Syntax**

Legend.Columns

### **Arguments**

None

### **Returned Value**

Long            None            Number of columns.

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.Columns = 10  
mURL = mLegend.URL  
Response.write mURL
```

## **Legend.Display property**

### **Description**

The Legend.Display property turns the legend on or off.

### **Syntax**

Legend.Display

### **Arguments**

None

### **Returned Value**

Boolean      Turns legend on or off.

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.Display = True  
mURL = mLegend.URL  
Response.write mURL
```

## **Legend.Font property**

### **Description**

The Legend.Font property returns or sets the title's font name.

### **Syntax**

Legend.Font

### **Arguments**

None

### **Returned Value**

String            None            Title's font name.

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.Font = "Arial"  
mURL = mLegend.URL  
Response.write mURL
```

## **Legend.Height property**

### **Description**

The Legend.Height property contains the legend height in pixels.

### **Syntax**

Legend.Height

### **Arguments**

None

### **Returned Value**

Long            None            Legend height in pixels.

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.Height = 10  
mURL = mLegend.URL  
Response.write mURL
```

## **Legend.LayerFontSize property**

### **Description**

The Legend.LayerFontSize property contains the layer name's font size.

### **Syntax**

Legend.LayerFontSize

### **Arguments**

None

### **Returned Value**

Long              None              Layer's font size.

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.LayerFontSize = 10  
mURL = mLegend.URL  
Response.write mURL
```

## **Legend.ReverseOrder property**

### **Description**

The Legend.ReverseOrder property reverses the order of the layers.

### **Syntax**

Legend.ReverseOrder

### **Arguments**

None

### **Returned Value**

Boolean      Reverses order of layers.

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.ReverseOrder = True  
mURL = mLegend.URL  
Response.Write mURL
```

## **Legend.SplitText property**

### **Description**

The Legend.SplitText property displays in the bottom of every column that has been split for the valuemap.

### **Syntax**

Legend.SplitText

### **Arguments**

None

### **Returned Value**

String	None	Displays in bottom of every column that has been split for valuemap.
--------	------	--

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.SplitText = "Split text"  
mURL = mLegend.URL  
Response.write mURL
```

## **Legend.SwatchHeight property**

### **Description**

The Legend.SwatchHeight property contains the swatch height.

### **Syntax**

Legend.SwatchHeight

### **Arguments**

None

### **Returned Value**

Long            None            Swatch height.

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.SwatchHeight = 10  
mURL = mLegend.URL  
Response.write mURL
```

## **Legend.SwatchWidth property**

### **Description**

The Legend.SwatchWidth property contains the swatch width.

### **Syntax**

Legend.Swatch Width

### **Arguments**

None

### **Returned Value**

Long            None            Swatch width.

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.SwatchWidth = 10  
mURL = mLegend.URL  
Response.write mURL
```

## **Legend.Title property**

### **Description**

The Legend.Title property contains the title of the entire legend.

### **Syntax**

Legend.Title

### **Arguments**

None

### **Returned Value**

String            None            Title.

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.Title = "Legend"  
mURL = mLegend.URL  
Response.write mURL
```

### **Note**

Also refer to samples:

    Sample2.asp

## **Legend.TitleFontSize property**

### **Description**

The Legend.TitleFontSize property contains the font size of the title to be placed on the legend.

### **Syntax**

Legend.TitleFontSize

### **Arguments**

None

### **Returned Value**

Long              None              Title's font size.

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.TitleFontSize = 10  
mURL = mLegend.URL  
Response.write mURL
```

### **Note**

Also refer to samples:

  Sample2.asp

## **Legend.Transcolor property**

### **Description**

The Legend.Transcolor property contains transparency information for the legend.

### **Syntax**

Legend.Transcolor

### **Arguments**

None

### **Returned Value**

imsColor                  None                  mLegend.Transcolor

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.Transcolor = 255  
mURL = mLegend.URL  
Response.write mURL
```

## **Legend.URL property**

### **Description**

The Legend.URL property contains the HTTP address to the legend graphic.

### **Syntax**

Legend.URL

### **Arguments**

None

### **Returned Value**

String            None            Description.

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mURL = mLegend.URL  
Response.write mURL
```

### **Note**

Also refer to samples:

Sample2.asp

## **Legend.ValueFontSize property**

### **Description**

The Legend.ValueFontSize property contains the font size for unique values of layers.

### **Syntax**

Legend.ValueFontSize

### **Arguments**

None

### **Returned Value**

Long            None            Value font size.

### **Example**

```
Dim mLegend  
Dim mURL  
Set mLegend = Server.CreateObject("aims.Legend")  
mLegend.ValueFontSize = 10  
mURL = mLegend.URL  
Response.write mURL
```

## **Legend.Width property**

### **Description**

The Legend.Width property returns or sets the legend width in pixels.

### **Syntax**

Legend.Width

### **Arguments**

None

### **Returned Value**

Long            None            Legend width in pixels.

### **Example**

Dim mLegend

Dim mURL

```
Set mLegend = Server.CreateObject("aims.Legend")
```

```
mLegend.Width = 10
```

```
mURL = mLegend.URL
```

```
Response.write mURL
```

## **LineObject.Id property**

### **Description**

The LineObject.Id property returns or sets a string identifier for the line.

### **Syntax**

LineObject.Id

### **Arguments**

None

### **Returned Value**

String            “Line”            Line’s id.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mLineObject  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSSMapService"  
Set mLineObject = Server.CreateObject( "aims.LineObject" )  
mId = mLineObject.Id  
Response.write mId
```

## **LineObject.Name property**

### **Description**

The LineObject.Name property returns or sets the line's name.

### **Syntax**

LineObject.Name

### **Arguments**

None

### **Returned Value**

String            “Line”            Line’s name.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mLineObject  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSSMapService"  
Set mLineObject = Server.CreateObject( "aims.LineObject" )  
mLineObject.Name = "The Line"  
Response.write mLineObject.Name
```

## **LineObject.Parts property**

### **Description**

The LineObject.Parts property contains parts of the line. Each part is a simple line.

### **Syntax**

LineObjects.Parts

### **Arguments**

None

### **Returned Value**

Parts            None            Parts collection.

### **Example**

```
Dim mAclMSConnector  
Dim mMap  
Dim mLineObject  
Dim mParts  
Dim mPointObject  
Dim mPoints  
Dim al  
Set mAclMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mAclMSConnector.ServerName = "IMSServer"  
mAclMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mAclMSConnector, "IMSSMapService"  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.visible = True  
Set mPoints = Server.CreateObject("aims.Points")  
Set mPointObject = Server.CreateObject("aims.PointObject")  
mPointObject.x = -125.0  
mPointObject.y = 45.0  
mPoints.Add mPointObject  
Set mLineObject = Server.CreateObject("aims.LineObject")  
Set mParts = mLineObject.Parts  
mParts.Add mPoints  
al.Add mLineObject, 1
```

## **LineObject.Symbol property**

### **Description**

The LineObject.Symbol property specifies how to depict the line.

### **Syntax**

LineObject.Symbol

### **Arguments**

None

### **Returned Value**

Object	SimpleLineSymbol	Any line symbol.
--------	------------------	------------------

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mLineObject  
Dim mParts  
Dim mPointObject  
Dim mPoints  
Dim al  
  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.visible = True  
  
Set mPoints = Server.CreateObject("aims.Points")  
Set mPointObject = Server.CreateObject("aims.PointObject")  
mPointObject.x = -125.0  
mPointObject.y = 45.0  
mPoints.Add mPointObject  
  
Set mLineObject = Server.CreateObject("aims.LineObject")  
mLineObject.Parts.Add mPoints
```

```
mLineObject.Symbol.Color = 255  
al.Add mLineObject, 1  
mMap.Layers.Add al  
mMap.Refresh
```

#### **See Also**

[HashLineSymbol](#)

[SimpleLineSymbol](#)

## **Map.BackColor property**

### **Description**

The Map.BackColor property returns or sets the background color of the map.

### **Syntax**

Map.BackColor

### **Arguments**

None

### **Returned Value**

Long	White	Background color of the map. Use imsColor and imsGlobalColor constants.
------	-------	---

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSSMapService"  
mMap.BackColor = 128  
mMap.Refresh
```

### **See Also**

imsColor

imsGlobalColor

### **Note**

Also refer to samples:

Sample1.asp

## **Map.CenterAt method**

### **Description**

The Map.CenterAt method sets map center to the specified point.

### **Syntax**

Map.CenterAt(X, Y)

### **Arguments**

X	Long	X coordinate of the center in pixels.
Y	Long	Y coordinate of the center in pixels.

### **Returned Value**

None

### **Examples**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
mMap.CenterAt 100, 100  
mMap.Refresh
```

### **Note**

Also refer to samples:

Sample3.asp

## **Map.DoPan method**

### **Description**

The Map.DoPan method shifts the current extent to the specified direction.

### **Syntax**

Map.DoPan (direction as Long, step as Double)

### **Arguments**

direction	Long	None	Specifies direction. Use imsDirection constant.
step	Double	None	Specifies the step for pan.

### **Returned Value**

None

### **Example**

```
Dim mArcIMSConnector
Dim mMap
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject( "aims.Map" )
mMap.InitMap mArcIMSConnector, "IMSMAPService"
mMap.DoPan imsDirection.imsNorth, 0.5
mMap.Refresh
```

### **See Also**

imsDirection

## **Map.DoZoom method**

### **Description**

The Map.DoZoom method controls zooming of the map.

### **Syntax**

Map.DoZoom(ScaleFactor)

### **Arguments**

ScaleFactor      Long      None      Scale factor > 0 for zoom in. Scale factor < 0 for zoom out.

### **Returned Value**

None

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
mMap.DoZoom 1  
mMap.Refresh
```

### **See Also**

[Map.DoZoomToExtent](#)  
[Map.DoZoomToFullExtent](#)  
[Map.DoZoomToFeatures](#)  
[Map.DoPan](#)

### **Note**

Also refer to samples:  
[Sample3.asp](#)

## Map.DoZoomToExtent method

### Description

The Map.DoZoomToExtent method sets the specified extent as current.

### Syntax

Map.DoZoomToExtent(Extent)

### Arguments

Extent	Envelope	None	The required extent.
--------	----------	------	----------------------

### Returned Value

Boolean	True indicates success. False indicates an error.
---------	---

### Example

```
Dim mArcIMSConnector
Dim mMap
Dim mExtent
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
Set mMap = Server.CreateObject( "aims.Map" )
Set mExtent = Server.CreateObject( "aims.Envelope" )
mMap.InitMap mArcIMSConnector, "IMSMAPService"
mExtent.XMin = 112
mExtent.YMin = -8
mExtent.XMax = 203
mExtent.YMax = 64
mMap.DoZoomToExtent mExtent
mMap.Refresh
```

### See Also

[Map.DoZoomTo](#)  
[Map.DoZoomToFullExtent](#)  
[Map.DoZoomToFeatures](#)  
[Map.DoPan](#)

## **Map.DoZoomToFeatures method**

### **Description**

The Map.DoZoomToFeatures method sets the extent of the specified feature as current.

### **Syntax**

Map.DoZoomToFeatures(ActiveLayer, FieldName, Values)

### **Arguments**

ActiveLayer	FeatureLayer	None	Active layer.
FieldName	String	None	Specifies the field name.
Values	String	None	Values for search delimited by comma.

### **Returned Value**

Boolean      True indicates success. False indicates an error.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
mMap.DoZoomToFeatures mMap.Layers.Item(2), "CNTRY_NAME", "USA", "Russia"  
mMap.Refresh
```

### **See Also**

[Map.DoZoomTo](#)  
[Map.DoZoomToExtent](#)  
[Map.DoZoomToFullExtent](#)  
[Map.DoPan](#)

## **Map.DoZoomToFullExtent method**

### **Description**

The Map.DoZoomToFullExtent method sets the initial extent as current.

### **Syntax**

Map.DoZoomToFullExtent()

### **Arguments**

None

### **Returned Value**

Boolean      True indicates success. False indicates an error.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
mMap.DoZoomToFullExtent()  
mMap.Refresh
```

### **See Also**

[Map.DoZoomTo](#)

[Map.DoZoomToExtent](#)

[Map.DoZoomToFeatures](#)

[Map.DoPan](#)

## **Map.Extent property**

### **Description**

The Map.Extent property sets the current extent of the map.

### **Syntax**

Map.Extent

### **Arguments**

None

### **Returned Value**

Envelope      Full extent      Current extent of the map.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mExtent  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set mExtent = mMap.Extent
```

### **See Also**

Envelope

### **Note**

Also refer to samples:

- Sample2.asp
- Sample4.asp
- Sample12.asp

## **Map.FromMapPoint method**

### **Description**

The Map.FromMapPoint method converts coordinates from database units to pixel units.

### **Syntax**

PointObject Map.FromMapPoint(X,Y)

### **Arguments**

X	Double	X coordinate in database units.
Y	Double	Y coordinate in database units.

### **Returned Value**

PointObject      None      PointObject contains converted coordinates in pixel units.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mPointObject  
  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
  
Set mPointObject = mMap.FromMapPoint(122, 37)  
Response.write mPointObject.x & ", " & mPointObject.y
```

### **See Also**

[Map.ToMapPoint](#)

## **Map.GetImageAsURL method**

### **Description**

The Map.GetImageAsURL method returns the HTTP address of the generated image.

### **Syntax**

Map.GetImageAsURL()

### **Arguments**

None

### **Returned Value**

String            None            Address of the picture.

### **Examples**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
mUrl = mMap.GetImageAsUrl()  
Response.write mUrl
```

### **Note**

Also refer to samples:

Sample1.asp

## **Map.Height property**

### **Description**

The Map.Height property sets the height of the map in pixels.

### **Syntax**

Map.Height

### **Arguments**

None

### **Returned Value**

Long            100            Height of the map in pixels.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSSMapService"  
mMap.Height = 300
```

### **See Also**

[Map.Width](#)

### **Note**

Also refer to samples:

[Sample1.asp](#)

## Map.InitMap method

### Description

The Map.InitMap method initializes the service, and loads the necessary renderer, recordset, and geocoding data.

### Syntax

Map.InitMap(Connector, ServiceName, LoadRenderer, LoadRecordset, LoadGeocode)

### Arguments

Connector	ArcIMSConnector	Nothing	ArcIMSConnector object which contains all connection parameters.
ServiceName	String	“”	Name of the service.
LoadRenderer	Boolean	True	Optional parameter. If parameter is false, the layers will not work with Recordset and Recordset property of all featureclass layers will be Nothing.
LoadGeocode	Boolean	True	Optional parameter. If parameter is false, the connector won't send additional request to download geocoding data and AddressMatchInputs property of all featureclass layers will be Nothing.

### Returned Value

Boolean      True      True indicates success. False indicates an error. To get the last error use ArcIMSConnector.GetLastError() method.

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
mMap.Refresh
```

### See Also

[ArcIMSConnector](#)  
[ArcIMSConnector.GetLastError](#)

**Note**

Also refer to samples:  
Sample1.asp

## **Map.Layers property**

### **Description**

The Map.Layers property contains all layers defined in the service.

### **Syntax**

Map.Layers

### **Arguments**

None

### **Returned Value**

Layers            None            Layers collection.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mLayers  
  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mLayers = mMap.Layers  
For i = 1 to mLayers.Count  
Response.Write mLayers.Item(i).Name  
Next
```

### **See Also**

Layers

### **Note**

Also refer to samples:  
    Sample1.asp

## **Map.Legend property**

### **Description**

The Map.Legend property contains all legend parameters.

### **Syntax**

Map.Legend

### **Arguments**

None

### **Returned Value**

Legend            None            Legend of the map.

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mLegend
```

```
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )
```

```
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject( "aims.Map" )
```

```
mMap.InitMap mArcIMSConnector, "IMSSMapService"
```

```
Set mLegend = mMap.Legend
```

```
Response.write mLegend.URL
```

### **See Also**

Legend

### **Note**

Also refer to samples:

Sample2.asp

## **Map.Refresh method**

### **Description**

The Map.Refresh method refreshes the map.

### **Syntax**

Map.Refresh()

### **Arguments**

None

### **Returned Value**

Boolean      True indicates success. False indicates an error. To get last error description use ArcIMSConnector.GetLastError().

### **Example**

```
Dim mArcIMSConnector  
Dim mMap
```

```
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
mMap.Refresh()
```

### **See Also**

ArcIMSConnector.GetLastError

### **Note**

Also refer to samples:

Sample1.asp

## **Map.ToMapPoint method**

### **Description**

The Map.ToMapPoint method converts coordinates from pixel units to database units.

### **Syntax**

Map.ToMapPoint(X,Y)

### **Arguments**

X	Long	None	X coordinate in pixels.
Y	Long	None	Y coordinate in pixels.

### **Returned Value**

PointObject      None      PointObject contains converted coordinates in database units.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mPointObject  
  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
  
Set mMap = Server.CreateObject( "aims.Map" )  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
  
Set mPointObject = mMap.ToMapPoint(100, 100)  
Response.write mPointObject.x & "," & mPointObject.y
```

### **See Also**

[Map.FromMapPoint](#)

### **Note**

Also refer to samples:

[Sample7.asp](#)  
[Sample11.asp](#)

## **Map.Width property**

### **Description**

The Map.Width property sets the width of the map in pixels.

### **Syntax**

Map.Width

### **Arguments**

None

### **Returned Value**

Long            100            Width of the map in pixels.

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )
```

```
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject( "aims.Map" )
```

```
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
mMap.Width = 500
```

### **See Also**

Map.height

### **Note**

Also refer to samples:

Sample1.asp

## **NorthArrowObject.Angle property**

### **Description**

The NorthArrowObject.Angle property sets the angle in simple degrees at which to turn the north arrow.

### **Syntax**

NorthArrowObject.Angle

### **Arguments**

None

### **Returned Value**

Double            90            Angle at which to turn the north arrow.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim al  
Dim mNorthArrowObject  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.visible = true  
  
Set mNorthArrowObject = Server.CreateObject( "aims.NorthArrowObject" )  
  
mNorthArrowObject.Angle = 31  
mNorthArrowObject.x = 25  
mNorthArrowObject.y = 25  
mNorthArrowObject.Size = 15  
mNorthArrowObject.ArrowType = 2  
al.Add mNorthArrowObject, 0  
mMap.Layers.Add al  
mMap.Refresh
```

## **NorthArrowObject.Antialiasing property**

### **Description**

The NorthArrowObject.Antialiasing property turns antialiasing on/off. Antialiasing allows lines to appear smooth by coloring adjacent pixels in such a way as to give the edge of a line a more gradual fade to the background.

### **Syntax**

NorthArrowObject.Antialiasing

### **Arguments**

None

### **Returned Value**

Boolean      False      Turns antialiasing on/off.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim al  
Dim mNorthArrowObject  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.visible = true  
Set mNorthArrowObject = Server.CreateObject( "aims.NorthArrowObject" )  
mNorthArrowObject.Angle = 31  
mNorthArrowObject.Antialiasing = true  
mNorthArrowObject.x = 25  
mNorthArrowObject.y = 25  
mNorthArrowObject.Size = 15  
mNorthArrowObject.ArrowType = 2  
al.Add mNorthArrowObject, 0  
mMap.Layers.Add al  
mMap.Refresh
```

## **NorthArrowObject.ArrowType property**

### **Description**

The NorthArrowObject.ArrowType property sets the arrow type.

### **Syntax**

NorthArrowObject.ArrowType

### **Arguments**

None

### **Returned Value**

ArrowType      Type 1      Arrow type. Use imsArrowType constants.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim al  
Dim mNorthArrowObject  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.visible = true  
  
Set mNorthArrowObject = Server.CreateObject( "aims.NorthArrowObject" )  
  
mNorthArrowObject.Angle = 31  
mNorthArrowObject.x = 25  
mNorthArrowObject.y = 25  
mNorthArrowObject.Size = 15  
mNorthArrowObject.ArrowType = 2  
al.Add mNorthArrowObject, 0  
mMap.Layers.Add al  
mMap.Refresh
```

## **See Also**

[imsArrowType](#)

## **Note**

Also refer to samples:

[Sample9.asp](#)

## NorthArrowObject.Id property

### Description

The NorthArrowObject.Id property returns or sets a string identifier for the NorthArrowObject.

### Syntax

NorthArrowObject.Id

### Arguments

None

### Returned Value

String        “North Arrow”    NorthArrowObject’s id.

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim al  
Dim mNorthArrowObject  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.visible = true  
Set mNorthArrowObject = Server.CreateObject( "aims.NorthArrowObject" )  
mNorthArrowObject.Angle = 31  
mNorthArrowObject.x = 25  
mNorthArrowObject.y = 25  
mNorthArrowObject.Size = 15  
mNorthArrowObject.ArrowType = 2  
mNorthArrowObject.Id = "NARROW:1976"  
al.Add mNorthArrowObject, 0  
mMap.Layers.Add al  
mMap.Refresh
```

## **NorthArrowObject.Name property**

### **Description**

The NorthArrowObject.Name property sets the name.

### **Syntax**

NorthArrowObject.Name

### **Arguments**

None

### **Returned Value**

String        “NorthArrow”    NorthArrowObject’s name.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim al  
Dim mNorthArrowObject  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.visible = true  
Set mNorthArrowObject = Server.CreateObject("aims.NorthArrowObject")  
mNorthArrowObject.Angle = 31  
mNorthArrowObject.x = 25  
mNorthArrowObject.y = 25  
mNorthArrowObject.Size = 15  
mNorthArrowObject.ArrowType = 2  
mNorthArrowObject.Name = "NorthArrow 5"  
al.Add mNorthArrowObject, 0  
mMap.Layers.Add al  
mMap.Refresh
```

## NorthArrowObject.Outline property

### Description

The NorthArrowObject.Outline property sets the outline color.

### Syntax

NorthArrowObject.Outline

### Arguments

None

### Returned Value

imsColor                  None                  Outline color.

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim al  
Dim mNorthArrowObject  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSSMapService"  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.visible = true  
Set mNorthArrowObject = Server.CreateObject("aims.NorthArrowObject")  
mNorthArrowObject.Angle = 31  
mNorthArrowObject.x = 25  
mNorthArrowObject.y = 25  
mNorthArrowObject.Size = 15  
mNorthArrowObject.ArrowType = 2  
mNorthArrowObject.Outline = 255  
al.Add mNorthArrowObject, 0  
mMap.Layers.Add al  
mMap.Refresh
```

### See Also

imsColor

## **NorthArrowObject.Overlap property**

### **Description**

The NorthArrowObject.Overlap property sets the overlap trigger.

### **Syntax**

NorthArrowObject.Overlap

### **Arguments**

None

### **Returned Value**

Boolean      True      Overlap trigger.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim al  
Dim mNorthArrowObject  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.visible = true  
Set mNorthArrowObject = Server.CreateObject("aims.NorthArrowObject")  
mNorthArrowObject.Angle = 31  
mNorthArrowObject.x = 25  
mNorthArrowObject.y = 25  
mNorthArrowObject.Size = 15  
mNorthArrowObject.ArrowType = 2  
mNorthArrowObject.Overlap = False  
al.Add mNorthArrowObject, 0  
mMap.Layers.Add al  
mMap.Refresh
```

## **NorthArrowObject.Shadow property**

### **Description**

The NorthArrowObject.Shadow property sets the shadow color.

### **Syntax**

NorthArrowObject.Shadow

### **Arguments**

None

### **Returned Value**

imsColor                None                Shadow color.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim al  
Dim mNorthArrowObject  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.visible = true  
Set mNorthArrowObject = Server.CreateObject("aims.NorthArrowObject")  
mNorthArrowObject.Angle = 31  
mNorthArrowObject.x = 25  
mNorthArrowObject.y = 25  
mNorthArrowObject.Size = 15  
mNorthArrowObject.ArrowType = 2  
mNorthArrowObject.Shadow = 0  
al.Add mNorthArrowObject, 0  
mMap.Layers.Add al  
mMap.Refresh
```

### **See Also**

imsColor

## **NorthArrowObject.Size property**

### **Description**

The NorthArrowObject.Size property sets the arrow size.

### **Syntax**

NorthArrowObject.Size

### **Arguments**

None

### **Returned Value**

Long              30              Arrow size.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim al  
Dim mNorthArrowObject  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSSMapService"  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.visible = true  
Set mNorthArrowObject = Server.CreateObject( "aims.NorthArrowObject" )  
mNorthArrowObject.Angle = 31  
mNorthArrowObject.x = 25  
mNorthArrowObject.y = 25  
mNorthArrowObject.Size = 15  
mNorthArrowObject.ArrowType = 2  
al.Add mNorthArrowObject, 0  
mMap.Layers.Add al  
mMap.Refresh
```

### **Note**

Also refer to samples:

Sample9.asp

## **NorthArrowObject.Transparency property**

### **Description**

The NorthArrowObject.Transparency property sets the transparency for the north arrow.

### **Syntax**

NorthArrowObject.Transparency

### **Arguments**

None

### **Returned Value**

Double            1.0            Transparency coefficient

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim al  
Dim mNorthArrowObject  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSSMapService"  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.visible = true  
  
Set mNorthArrowObject = Server.CreateObject( "aims.NorthArrowObject" )  
  
mNorthArrowObject.Angle = 31  
mNorthArrowObject.x = 25  
mNorthArrowObject.y = 25  
mNorthArrowObject.Size = 15  
mNorthArrowObject.ArrowType = 2  
mNorthArrowObject.Transparency = 0.7  
al.Add mNorthArrowObject, 0
```

```
mMap.Layers.Add al
```

```
mMap.Refresh
```

**Note**

Also refer to samples:

Sample9.asp

## **NorthArrowObject.X property**

### **Description**

The NorthArrowObject.X property sets the X coordinate of arrow location.

### **Syntax**

NorthArrowObject.X

### **Arguments**

None

### **Returned Value**

Double            0            X coordinate.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim al  
Dim mNorthArrowObject  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.visible = true  
Set mNorthArrowObject = Server.CreateObject("aims.NorthArrowObject")  
mNorthArrowObject.Angle = 31  
mNorthArrowObject.x = 25  
mNorthArrowObject.y = 25  
mNorthArrowObject.Size = 15  
mNorthArrowObject.ArrowType = 2  
al.Add mNorthArrowObject, 0  
mMap.Layers.Add al  
mMap.Refresh
```

### **Note**

Also refer to samples:

Sample9.asp

## **NorthArrowObject.Y property**

### **Description**

The NorthArrowObject.Y property sets the Y coordinate of the arrow location.

### **Syntax**

NorthArrowObject.Y

### **Arguments**

None

### **Returned Value**

Double            0            Y coordinate location.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim al  
Dim mNorthArrowObject  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSSMapService"  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.visible = true  
Set mNorthArrowObject = Server.CreateObject( "aims.NorthArrowObject" )  
mNorthArrowObject.Angle = 31  
mNorthArrowObject.x = 25  
mNorthArrowObject.y = 25  
mNorthArrowObject.Size = 15  
mNorthArrowObject.ArrowType = 2  
al.Add mNorthArrowObject, 0  
mMap.Layers.Add al  
mMap.Refresh
```

### **Note**

Also refer to samples:

Sample9.asp

## Parts.Add method

### Description

The Parts.Add method adds a Points object to the Parts collection.

### Syntax

Parts.Add(Points)

### Arguments

Points	Points	None	Points object to add.
--------	--------	------	-----------------------

### Returned Value

None

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim mParts  
Dim mPoints  
Dim mPoint  
Dim al  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set mParts = Server.CreateObject("aims.Parts")  
Set mPoints = Server.CreateObject("aims.Points")  
Set mPoint = Server.CreateObject("aims.PointObject")  
mPoint.x = -125.00  
mPoint.y = 45.00  
mPoints.Add mPoint  
mParts.Add( mPoints )  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.visible = true  
al.Add mPoints, 1  
mMap.Layers.Add al  
mMap.Refresh
```

## **Parts.Clear method**

### **Description**

The Parts.Clear method removes all Points objects from the Parts collection.

### **Syntax**

Parts.Clear()

### **Arguments**

None

### **Returned Value**

None

### **Example**

```
Dim mParts  
Dim mPoints  
Set mParts = Server.CreateObject( "aims.Parts" )  
Set mPoints = Server.CreateObject( "aims.Points" )  
mParts.Add( mPoints )
```

## **Parts.Count property**

### **Description**

The Parts.Count property returns the number of Points objects in the Parts collection.

### **Syntax**

Parts.Count

### **Arguments**

None

### **Returned Value**

Long	None	Number of Points objects in the Parts collection.
------	------	---

### **Example**

```
Dim mParts
Dim mPoints
Set mParts = Server.CreateObject( "aims.Parts" )
Set mPoints = Server.CreateObject( "aims.Points" )
mParts.Add( mPoints )
mCount = mParts.Count
```

## **Parts.Insert method**

### **Description**

The Parts.Insert method inserts a Points object at the specified position.

### **Syntax**

Parts.Insert(index, Points)

### **Arguments**

index	Long	None	Points object position.
Points	Points	None	Points object to insert.

### **Returned Value**

None

### **Example**

```
Dim mParts  
Dim mPoints  
Set mParts = Server.CreateObject( "aims.Parts" )  
Set mPoints = Server.CreateObject( "aims.Points" )  
mParts.Insert( 1, mPoints )
```

## **Parts.Item method**

### **Description**

The Parts.Item method returns the Points object by index.

### **Syntax**

Parts.Item(index)

### **Arguments**

Index	Long	None	Points object position.
-------	------	------	-------------------------

### **Returned Value**

Points	None	Description
--------	------	-------------

### **Example**

```
Dim mParts
Dim mPoints
Set mParts = Server.CreateObject( "aims.Parts" )
Set mPoints = Server.CreateObject( "aims.Points" )
mParts.Add( mPoints )
Set mPoints = mParts.Item(1)
```

## **Parts.Remove method**

### **Description**

The Parts.Remove method removes the Points object from the collection.

### **Syntax**

Parts.Remove(Index)

### **Arguments**

Index	Long	None	Points object position.
-------	------	------	-------------------------

### **Returned Value**

None

### **Example**

```
Dim mParts  
Dim mPoints  
Set mParts = Server.CreateObject( "aims.Parts" )  
Set mPoints = Server.CreateObject( "aims.Points" )  
mParts.Add( mPoints )  
mParts.Remove(1)
```

## **Parts.Replace method**

### **Description**

The Parts.Replace method replaces the Points object.

### **Syntax**

Parts.Replace(index, Points)

### **Arguments**

index	Long	None	Objects position.
Points	Points	None	New Points object

### **Returned Value**

None

### **Example**

```
Dim mParts
Dim mPoints
Set mParts = Server.CreateObject( "aims.Parts" )
Set mPoints = Server.CreateObject( "aims.Points" )
mParts.Add( mPoints )
mParts.Replace(1, mPoints )
```

## **PointObject.Id property**

### **Description**

The PointObject.Id property returns or sets a string identifier for a PointObject.

### **Syntax**

PointObject.Id

### **Arguments**

None

### **Returned Value**

String            The PointObject's id. The default value is "Point".

### **Example**

```
Dim mPointObject  
Set mPointObject = Server.CreateObject( "aims.PointObject" )  
mId = mPointObject.Id
```

## **PointObject.Name property**

### **Description**

The PointObject.Name property returns or sets a string name for a PointObject.

### **Syntax**

PointObject.Name

### **Arguments**

None

### **Returned Value**

String      PointObject's name. The default value is "Point".

### **Example**

```
Dim mPointObject  
Set mPointObject = Server.CreateObject( "aims.PointObject" )  
mName = mPointObject.Name
```

## **PointObject.Symbol property**

### **Description**

The PointObject.Symbol returns the Symbol object of the PointObject. The symbol object type is either SimpleMarkerSymbol, TrueTypeMarkerSymbol, or RasterMarkerSymbol.

### **Syntax**

PointObject.Symbol

### **Arguments**

None

### **Returned Value**

Object                Any polygon symbol. The default object is SimpleMarkerSymbol.

### **Example**

```
Dim mPointObject  
Dim mSymbol  
Set mPointObject = Server.CreateObject( "aims.PointObject" )  
Set mSymbol = mPointObject.Symbol
```

### **See Also**

[TrueTypeMarkerSymbol](#)  
[SimpleMarkerSymbol](#)  
[RasterMarkerSymbol](#)

## **PointObject.X property**

### **Description**

The PointObject.X property returns or sets the X coordinate of the point. The coordinate units can be in pixel or map units.

### **Syntax**

PointObject.X

### **Arguments**

None

### **Returned Value**

Double        X coordinate. The default value is 0.

### **Example**

```
Dim mPointObject  
Set mPointObject = Server.CreateObject( "aims.PointObject" )  
mPointObject.Y = -37.32  
mPointObject.X = -122.7
```

### **See Also**

PointObject.Y

### **Note**

Also refer to samples:

Select\_and\_Highlight.asp  
Identify.asp

## **PointObject.Y property**

### **Description**

The PointObject.Y property returns or sets the Y coordinate of the point. The coordinate units can be in pixel or map units.

### **Syntax**

PointObject.Y

### **Arguments**

None

### **Returned Value**

Double        Y coordinate. The default value is 0.

### **Example**

```
Dim mPointObject  
Set mPointObject = Server.CreateObject( "aims.PointObject" )  
mPointObject.Y = -37.32  
mPointObject.X = -122.7
```

### **See Also**

PointObject.Y

### **Note**

Also refer to samples:  
Select\_and\_Highlight.asp  
Identify.asp

## Points.Add method

### Description

The Points.Add method adds a point to a Points collection.

### Syntax

Points.Add (Point)

### Arguments

Point              Point to add.

### Returned Value

None

### Example

```
Dim mPoints  
Dim mPointObject  
Set mPoints = Server.CreateObject( "aims.Points" )  
Set mPointObject = Server.CreateObject( "aims.PointObject" )  
mPoints.Add mPointObject
```

### See Also

[PointObject](#)

## **Points.Clear method**

### **Description**

The Points.Clear method removes all points from the Points collection.

### **Syntax**

Points.Clear()

### **Arguments**

None

### **Returned Value**

None

### **Example**

```
Dim mPoints  
Dim mPointObject  
Set mPoints = Server.CreateObject( "aims.Points" )  
Set mPointObject = Server.CreateObject( "aims.PointObject" )  
mPoints.Add mPointObject  
mPoints.Clear()
```

## Points.Insert method

### Description

The Points.Insert method inserts a point at the specified position in a points collection. The inserted point is placed before the point that was previously located at the specified position.

### Syntax

```
Points.Insert(index, Point)
```

### Arguments

index              Points position.

Point              Point to be inserted.

### Returned Value

None

### Example

```
Dim mPoints  
Dim mPointObject  
Set mPoints = Server.CreateObject( "aims.Points" )  
Set mPointObject = Server.CreateObject( "aims.PointObject" )  
mPoints.Insert 1, mPointObject
```

## Points.Item method

### Description

The Points.Item method returns a point by its index in a Points collection.

### Syntax

Points.Item(Index)

### Arguments

Index              Points position.

### Returned Value

PointObject        Requested point.

### Example

```
Dim mPoints  
Dim mPointObject  
Set mPoints = Server.CreateObject( "aims.Points" )  
Set mPointObject = Server.CreateObject( "aims.PointObject" )  
mPoints.Add (mPointObject)  
Set mPointObject = mPoints.Item(1)
```

### See Also

PointObject

## Points.Replace method

### Description

The Points.Replace method replaces a point in a Points collection.

### Syntax

Points.Replace (index, Point)

### Arguments

index              Points position.

Point              New point.

### Returned Value

None

### Example

```
Dim mPoints
Dim mPointObject
Set mPoints = Server.CreateObject( "aims.Points" )
Set mPointObject = Server.CreateObject( "aims.PointObject" )
Set newObject = Server.CreateObject( "aims.PointObject" )
mPoints.Add mPointObject
newObject.X = 5
mPoints.Replace 1, newObject
```

### See Also

[PointObject](#)

## PolygonObject.Id property

### Description

The PolygonObject.Id property returns or sets a string identifier for a PolygonObject.

### Syntax

PolygonObject.Id

### Arguments

None

### Returned Value

String         The PolygonObject's id. The default value is "Polygon".

### Example

```
dim mTestPt1  
dim mTestPt2  
dim mTestPt3  
dim mTestPt4  
  
set TestPt1 = CreateObject("aims.PointObject")  
set TestPt2 = CreateObject("aims.PointObject")  
set TestPt3 = CreateObject("aims.PointObject")  
set TestPt4 = CreateObject("aims.PointObject")  
  
mTestPt1.X = 100  
mTestPt1.Y = 100  
mTestPt2.X = 200  
mTestPt2.Y = 200  
mTestPt3.X = 100  
mTestPt3.Y = 200  
mTestPt4.X = 200  
mTestPt4.Y = 100  
  
dim mPoints  
set mPoints = Server.CreateObject("aims.Points")
```

```
mPoints.Add mTestPt1  
mPoints.Add mTestPt2  
mPoints.Add mTestPt3  
mPoints.Add mTestPt4  
  
dim mPolygonObject  
dim mSymbol  
set mPolygonObject = Server.CreateObject( "aims.PolygonObject" )  
mPolygonObject.Id = "Polygon1"  
mPolygonObject.Name = "Two Triangles"  
set mSymbol = mPolygonObject.Symbol  
mPolygonObject.Parts.Add mPoints  
  
mResult = AcetateLayer1.Add( mPolygonObject, 0)
```

## PolygonObject.Name property

### Description

The PolygonObject.Name property returns or sets a string name for a PolygonObject.

### Syntax

PolygonObject.Name

### Arguments

None

### Returned Value

String      PolygonObject's name. The default value is "Polygon".

### Example

```
dim mPolygonObject  
dim mSymbol  
set mPolygonObject = Server.CreateObject( "aims.PolygonObject" )  
mPolygonObject.Id = "Polygon1"  
mPolygonObject.Name = "Two Triangles"  
set mSymbol = mPolygonObject.Symbol  
  
mResult = AcetateLayer1.Add( mPolygonObject, 0)
```

## PolygonObject.Parts property

### Description

The PolygonObject.Parts property returns the Parts object of a PolygonObject. The Parts collection contains parts of the polygon. Each part (which is a Points collection) in a Parts collection is a single polygon.

### Syntax

PolygonObject.Parts

### Arguments

None

### Returned Value

Parts              PolygonObject's Parts collection.

### Example

```
Dim mPolygonObject  
Dim mParts  
Set mPolygonObject = Server.CreateObject( "aims.PolygonObject" )  
Set mParts = mPolygonObject.Parts
```

### See Also

Parts

Points

## PolygonObject.Symbol property

### Description

The PolygonObject.Symbol property returns the Symbol object of the PolygonObject. The symbol object type is either SimplePolygonSymbol, RasterFillSymbol, or GradientFillSymbol.

### Syntax

```
PolygonObject.Symbol
```

### Arguments

None

### Returned Value

Object                Any polygon symbol. The default object is SimplePolygonSymbol.

### Example

```
dim mTestPt1
```

```
dim mTestPt2
```

```
dim mTestPt3
```

```
dim mTestPt4
```

```
set TestPt1 = CreateObject("aims.PointObject")
```

```
set TestPt2 = CreateObject("aims.PointObject")
```

```
set TestPt3 = CreateObject("aims.PointObject")
```

```
set TestPt4 = CreateObject("aims.PointObject")
```

```
mTestPt1.X = 100
```

```
mTestPt1.Y = 100
```

```
mTestPt2.X = 200
```

```
mTestPt2.Y = 200
```

```
mTestPt3.X = 100
```

```
mTestPt3.Y = 200
```

```
mTestPt4.X = 200
```

```
mTestPt4.Y = 100
```

```
dim mPoints
```

```
set mPoints = Server.CreateObject("aims.Points")
```

```
mPoints.Add mTestPt1  
mPoints.Add mTestPt2  
mPoints.Add mTestPt3  
mPoints.Add mTestPt4  
  
dim mPolygonObject  
dim mSymbol  
set mPolygonObject = Server.CreateObject( "aims.PolygonObject" )  
mPolygonObject.Id = "Polygon1"  
mPolygonObject.Name = "Two Triangles"  
set mSymbol = mPolygonObject.Symbol  
mPolygonObject.Parts.Add mPoints  
  
mResult = AcetateLayer1.Add( mPolygonObject, 0)
```

### See Also

[SimplePolygonSymbol](#)  
[RasterFillSymbol](#)  
[GradientFillSymbol](#)

## RasterFillSymbol.Antialiasing property

### Description

The RasterFillSymbol.Antialiasing property returns or sets a Boolean value indicating whether or not antialiasing will be used while rendering a RasterFillSymbol. Antialiasing allows lines to appear smooth by coloring adjacent pixels in such a way as to give the edge of a line a more gradual fade to the background.

### Syntax

RasterFillSymbol.Antialising

### Arguments

None

### Returned Value

Boolean      True indicates that Antialiasing is on. False means that it is turned off. The default value is False.

### Example

```
Dim mRasterFillSymbol  
Set mRasterFillSymbol = Server.CreateObject("aims.RasterFillSymbol")  
mRasterFillSymbol.Antialising = True
```

## RasterFillSymbol.Boundary property

### Description

The RasterFillSymbol.Boundary property returns or sets a Boolean value indicating whether or not a RasterFillSymbol's boundary will be drawn.

### Syntax

RasterFillSymbol.Boundary

### Arguments

None

### Returned Value

Boolean      True indicates that the boundary will be drawn. The default value is False

### Example

```
Dim mRasterFillSymbol  
Set mRasterFillSymbol = Server.CreateObject("aims.RasterFillSymbol")  
mRasterFillSymbol.Boundary = True
```

## RasterFillSymbol.Clone method

### Description

The RasterFillSymbol.Clone method clones the RasterFillSymbol.

### Syntax

RasterFillSymbol.Clone ()

### Arguments

None

### Returned Value

RasterFillSymbol      Cloned symbol.

### Example

```
dim mClone  
dim mRasterFillSymbol  
Set mRasterFillSymbol = Server.CreateObject("aims.RasterFillSymbol")  
Set mClone = mRasterFillSymbol.Clone()
```

## RasterFillSymbol.Image property

### Description

The RasterFillSymbol.Image property returns or sets the name of the image that will be used for the RasterFillSymbol. This could be a path to the image file. The ArcIMS Spatial Server reads this path to locate the image file, so the path will NOT be a URL, but a normal file system path.

### Syntax

RasterFillSymbol.Image

### Arguments

None

### Returned Value

String        The path and name of the image. The path to the image file is read by the ArcIMS Spatial Server.

### Example

```
Dim mRasterFillSymbol  
Set mRasterFillSymbol = Server.CreateObject("aims.RasterFillSymbol")  
mRasterFillSymbol.Image = "C:\Temp\Picture.gif"
```

## RasterFillSymbol.Overlap property

### Description

The RasterFillSymbol.Overlap property returns or sets a Boolean value indicating whether or not labels will overlap a RasterFillSymbol.

### Syntax

RasterFillSymbol.Overlap

### Arguments

None

### Returned Value

Boolean      True indicates that labels will be allowed to overlap the RasterFillSymbol. False indicates that labels will not overlap the symbol. The default value is True.

### Example

```
Dim mRasterFillSymbol  
Set mRasterFillSymbol = Server.CreateObject("aims.RasterFillSymbol")  
mRasterFillSymbol.Overlap = False
```

## RasterFillSymbol.Transparency property

### Description

The RasterFillSymbol.Transparency property returns or sets the transparency coefficient for the RasterFillSymbol. Lower numbers will display the symbol with greater transparency.

### Syntax

RasterFillSymbol.Transparency

### Arguments

None

### Returned Value

Double Transparency coefficient. The default value is 1.0.

### Example

```
Dim mRasterFillSymbol  
Set mRasterFillSymbol = Server.CreateObject("aims.RasterFillSymbol")  
mRasterFillSymbol.Transparency = 1.0
```

## RasterFillSymbol.URL property

### Description

The RasterFillSymbol.URL property returns or sets the URL to the image that will be used to fill a polygon feature. The client reads the URL to get the image for the RasterFillSymbol.

### Syntax

RasterFillSymbol.URL

### Arguments

None

### Returned Value

String      The URL string to an image.

### Example

```
Dim mRasterFillSymbol  
Set mRasterFillSymbol = Server.CreateObject("aims.RasterFillSymbol")  
mRasterFillSymbol.URL = "http://www.esri.com/Picture.gif"
```

## RasterMarkerSymbol.Antialiasing property

### Description

The RasterMarkerSymbol.Antialiasing property returns or sets a Boolean value indicating whether or not antialiasing will be used while rendering a RasterMarkerSymbol. Antialiasing allows lines to appear smooth by coloring adjacent pixels in such a way as to give the edge of a line a more gradual fade to the background.

### Syntax

RasterMarkerSymbol.Antialiasing

### Arguments

None

### Returned Value

Boolean      Turns antialiasing on/off. The default value is False.

### Example

```
Dim mRasterMarkerSymbol  
Set mRasterMarkerSymbol = Server.CreateObject("aims.RasterMarkerSymbol")  
mRasterMarkerSymbol.Antialiasing = True
```

## RasterMarkerSymbol.Clone method

### Description

The RasterMarkerSymbol.Clone method clones the RasterMarkerSymbol.

### Syntax

RasterMarkerSymbol.Clone ()

### Arguments

None

### Returned Value

RasterMarkerSymbol      Cloned symbol.

### Example

```
dim mClone  
dim mRasterMarkerSymbol  
Set mRasterMarkerSymbol = Server.CreateObject("aims.RasterMarkerSymbol")  
Set mClone = mRasterMarkerSymbol.Clone()
```

## RasterMarkerSymbol.HotSpotX property

### Description

The RasterMarkerSymbol.HotSpotX property returns or sets the X location where an image is placed in relation to a point. HotSpotX is left coordinate. HotSpotX is always positive and is measured in pixels.

### Syntax

RasterMarkerSymbol.HotSpotX

### Arguments

None

### Returned Value

Long            X coordinate.

### Example

```
Dim mRasterMarkerSymbol  
Set mRasterMarkerSymbol = Server.CreateObject("aims.RasterMarkerSymbol")  
mRasterMarkerSymbol.HotSpotX = 10
```

### See Also

RasterMarkerSymbol.HotSpotY

## RasterMarkerSymbol.HotSpotY property

### Description

The RasterMarkerSymbol.HotSpotY property returns or sets the Y location where an image is placed in relation to a point. HotSpotY is the top coordinate. HotSpotY is always positive and is measured in pixels.

### Syntax

RasterMarkerSymbol.HotSpotY

### Arguments

None

### Returned Value

Long            Y coordinate.

### Example

```
Dim mRasterMarkerSymbol  
Set mRasterMarkerSymbol = Server.CreateObject("aims.RasterMarkerSymbol")  
mRasterMarkerSymbol.HotSpotY = 10
```

### See Also

RasterMarkerSymbol.HotSpotX

## RasterMarkerSymbol.Image property

### Description

The RasterMarkerSymbol.Image property returns or sets the name of the image that will be used symbolize a point feature. This could be a path to the image file. The ArcIMS Spatial Server reads this path.

### Syntax

RasterMarkerSymbol.Image

### Arguments

None

### Returned Value

String      The path and name of the image. The path to the image file is read by the ArcIMS Spatial Server.

### Example

```
Dim mRasterMarkerSymbol  
Set mRasterMarkerSymbol = Server.CreateObject("aims.RasterMarkerSymbol")  
mRasterMarkerSymbol.Image = "C:\Temp\Picture.gif"
```

## RasterMarkerSymbol.Overlap property

### Description

The RasterMarkerSymbol.Overlap property returns or sets a Boolean value indicating whether or not labels will overlap a RasterMarkerSymbol.

### Syntax

RasterMarkerSymbol.Overlap

### Arguments

None

### Returned Value

Boolean      True indicates that labels will be allowed to overlap the RasterMarkerSymbol.  
False indicates that labels will not overlap the symbol. The default value is True.

### Example

```
Dim mRasterMarkerSymbol  
Set mRasterMarkerSymbol = Server.CreateObject("aims.RasterMarkerSymbol")  
mRasterMarkerSymbol.Overlap = False
```

## RasterMarkerSymbol.Shadow property

### Description

The RasterMarkerSymbol.Shadow property returns or sets a color constant for the shadow color of a RasterMarkerSymbol.

### Syntax

RasterMarkerSymbol.Shadow

### Arguments

None

### Returned Value

imsColor      Shadow color constant. The default value is imsBlack (0).

### Example

```
Dim mRasterMarkerSymbol  
Set mRasterMarkerSymbol = Server.CreateObject("aims.RasterMarkerSymbol")  
mRasterMarkerSymbol.Shadow = imsColor.imsBlack
```

## RasterMarkerSymbol.SizeX property

### Description

The RasterMarkerSymbol.SizeX property returns or sets the X width of the RasterMarkerSymbol bitmap. SizeX is always positive and is measured in pixels.

### Syntax

RasterMarkerSymbol.SizeX

### Arguments

None

### Returned Value

Long            X Width in pixels.

### Example

```
Dim mRasterMarkerSymbol  
Set mRasterMarkerSymbol = Server.CreateObject("aims.RasterMarkerSymbol")  
mRasterMarkerSymbol.SizeX= 10
```

### See Also

RasterMarkerSymbol.SizeY

## RasterMarkerSymbol.SizeY property

### Description

The RasterMarkerSymbol.SizeY property returns or sets the Y height of the RasterMarkerSymbol bitmap. SizeY is always positive and is measured in pixels.

### Syntax

RasterMarkerSymbol.SizeY

### Arguments

None

### Returned Value

Long            Y Height in pixels.

### Example

```
Dim mRasterMarkerSymbol  
Set mRasterMarkerSymbol = Server.CreateObject("aims.RasterMarkerSymbol")  
mRasterMarkerSymbol.SizeY = 10
```

### See Also

RasterMarkerSymbol.SizeX

## RasterMarkerSymbol.Transparency property

### Description

The RasterMarkerSymbol.Transparency property returns or sets the transparency coefficient for the RasterMarkerSymbol. Lower numbers will display the symbol with greater transparency.

### Syntax

RasterMarkerSymbol.Transparency

### Arguments

None

### Returned Value

Double Transparency coefficient. The default value is 1.0.

### Example

```
Dim mRasterMarkerSymbol  
Set mRasterMarkerSymbol = Server.CreateObject("aims.RasterMarkerSymbol")  
mRasterMarkerSymbol.Transparency = 1.0
```

## RasterShieldSymbol.Antialiasing property

### Description

The RasterShieldSymbol.Antialiasing property returns or sets a Boolean value indicating whether or not antialiasing will be used while rendering the RasterShieldSymbol. Antialiasing allows lines to appear smooth by coloring adjacent pixels in such a way as to give the edge of a line a more gradual fade to the background.

### Syntax

RasterShieldSymbol.Antialiasing

### Arguments

None

### Returned Value

Boolean      Turns antialiasing on/off. The default value is False.

### Example

```
Dim mRasterShieldSymbol  
Set mRasterShieldSymbol = Server.CreateObject("aims.RasterShieldSymbol")  
mRasterShieldSymbol.Antialiasing = True
```

## RasterShieldSymbol.Boundary property

### Description

The RasterShieldSymbol.Boundary property returns or sets a Boolean value indicating whether or not a boundary will be rendered for a RasterShieldSymbol.

### Syntax

RasterShieldSymbol.Boundary

### Arguments

None

### Returned Value

Boolean      Determines if a boundary will be drawn. The default value is False.

### Example

```
dim mRasterShieldSymbol  
set mRasterShieldSymbol = Server.CreateObject("aims.RasterShieldSymbol")  
mRasterShieldSymbol.Boundary = True
```

## RasterShieldSymbol.Clone method

### Description

The RasterShieldSymbol.Clone method clones the RasterShieldSymbol.

### Syntax

RasterShieldSymbol.Clone ()

### Arguments

None

### Returned Value

RasterShieldSymbol      Cloned symbol.

### Example

```
dim mClone  
dim mRasterShieldSymbol  
Set mRasterShieldSymbol = Server.CreateObject("aims.RasterShieldSymbol")  
Set mClone = mRasterShieldSymbol.Clone()
```

## RasterShieldSymbol.Font property

### Description

The RasterShieldSymbol.Font property returns or sets the name of the font used with the RasterShieldSymbol.

### Syntax

RasterShieldSymbol.Font

### Arguments

None

### Returned Value

String            Font name. The default value is “default”.

### Example

```
Dim mRasterShieldSymbol  
Set mRasterShieldSymbol = Server.CreateObject("aims.RasterShieldSymbol")  
mRasterShieldSymbol.Font = "Arial"
```

## RasterShieldSymbol.FontColor property

### Description

The RasterShieldSymbol.FontColor property returns or sets a font color constant for the font used with a RasterShieldSymbol.

### Syntax

RasterShieldSymbol.Color

### Arguments

None

### Returned Value

imsColor      Font color. The default value is imsBlack (0).

### Example

```
Dim mRasterShieldSymbol  
Set mRasterShieldSymbol = Server.CreateObject("aims.RasterShieldSymbol")  
mRasterShieldSymbol.Color = imsColor.imsRed
```

## **RasterShieldSymbol.FontSize property**

### **Description**

The RasterShieldSymbol.FontSize property returns or sets the font size for the font used with a RasterShieldSymbol.

### **Syntax**

RasterShieldSymbol.FontSize

### **Arguments**

None

### **Returned Value**

Long            Font size. The default value is 12.

### **Example**

```
Dim mRasterShieldSymbol  
Set mRasterShieldSymbol = Server.CreateObject("aims.RasterShieldSymbol")  
mRasterShieldSymbol.FontSize = 10
```

## RasterShieldSymbol.FontStyle property

### Description

The RasterShieldSymbol.FontStyle returns or sets a font style constant for the font used with a RasterShieldSymbol. The possible values are:

- 0 = imsRegular
- 1 = imsBold
- 2 = imsItalic
- 3 = imsUnderline
- 4 = imsOutline

### Syntax

RasterShieldSymbol.FontStyle

### Arguments

None

### Returned Value

imsFontStyle      Font style. The default value is imsRegular (0).

### Example

```
Dim mRasterShieldSymbol  
Set mRasterShieldSymbol = Server.CreateObject("aims.RasterShieldSymbol")  
mRasterShieldSymbol.FontStyle = imsFontStyle.imsBold
```

## RasterShieldSymbol.Image property

### Description

The RasterShieldSymbol.Image property returns or sets the name of the image that will be used for the RasterShieldSymbol. This could be a path to the image file. The ArcIMS Spatial Server reads this path.

### Syntax

RasterShieldSymbol.Image

### Arguments

None

### Returned Value

String      The path and name of the image. The path to the image file is read by the ArcIMS Spatial Server.

### Example

```
Dim mRasterShieldSymbol  
Set mRasterShieldSymbol = Server.CreateObject("aims.RasterShieldSymbol")  
mRasterShieldSymbol.Image = "C:\Temp\Picture.gif"
```

## RasterShieldSymbol.LabelMode property

### Description

The RasterShieldSymbol.LabelMode property returns or sets a LabelMode constant that determines what labels are drawn in the RasterShieldSymbol. The possible values are:

0 = imsFull

1 = imsNumericonly

‘Full’ means that the entire value of the label field will be displayed in the RasterShieldSymbol.

‘Numericonly’ means that only the numeric parts of the label field value will be displayed.

### Syntax

RasterShieldSymbol.LabelMode

### Arguments

None

### Returned Value

imsLabelMode    LabelMode constant. The default value is ‘imsFull’ (0).

### Example

```
Dim mRasterShieldSymbol  
Set mRasterShieldSymbol = Server.CreateObject("aims.RasterShieldSymbol")  
mRasterShieldSymbol.LabelMode = imsLabelMode.imsFull
```

### See Also

imsLabelMode

## RasterShieldSymbol.PrintMode property

### Description

The RasterShieldSymbol.PrintMode property returns or sets a PrintMode constant that determines how RasterShieldSymbol labels will be rendered. The possible values are:

- 0 = imsNone
- 1 = imsTitleCaps
- 2 = imsAllUpper
- 3 = imsAllLower

### Syntax

RasterShieldSymbol.PrintMode

### Arguments

None

### Returned Value

imsPrintMode Printing mode constant. The default value is imsNone (0).

### Example

```
Dim mRasterShieldSymbol  
Set mRasterShieldSymbol = Server.CreateObject("aims.RasterShieldSymbol")  
mRasterShieldSymbol.PrintMode = imsPrintMode.imsNone
```

### See Also

imsPrintMode

## RasterShieldSymbol.Shadow property

### Description

The RasterShieldSymbol.Shadow property returns or sets a color constant for the shadow color of a RasterShieldSymbol.

### Syntax

RasterShieldSymbol.Shadow

### Arguments

None

### Returned Value

imsColor      Shadow color constant. The default value is imsBlack (0).

### Example

```
Dim mRasterShieldSymbol  
Set mRasterShieldSymbol = Server.CreateObject("aims.RasterShieldSymbol")  
mRasterShieldSymbol.Shadow = imsColor.imsBlack
```

## **RasterShieldSymbol.TextPosition property**

### **Description**

The RasterShieldSymbol.TextPosition property returns or sets the “X,Y” location of the text on a map image.

### **Syntax**

RasterShieldSymbol.TextPosition

### **Arguments**

None

### **Returned Value**

String            X,Y position on the image. The default is the center of the image.

### **Example**

```
Dim mRasterShieldSymbol  
Set mRasterShieldSymbol = Server.CreateObject("aims.RasterShieldSymbol")  
mRasterShieldSymbol.TextPosition = "10,10"
```

## RasterShieldSymbol.Transparency property

### Description

The RasterShieldSymbol.Transparency property returns or sets the transparency coefficient for the RasterShieldSymbol. Lower numbers will display the symbol with greater transparency.

### Syntax

RasterShieldSymbol.Transparency

### Arguments

None

### Returned Value

Double Transparency coefficient. The default value is 1.0.

### Example

```
Dim mRasterShieldSymbol  
Set mRasterShieldSymbol = Server.CreateObject("aims.RasterShieldSymbol")  
mRasterShieldSymbol.Transparency = 1.0
```

## RasterShieldSymbol.URL property

### Description

The RasterShieldSymbol.URL property returns or sets the URL to the image that will be used for a RasterShieldSymbol. The client reads the URL to get the image for the RasterShieldSymbol.

### Syntax

RasterShieldSymbol.URL

### Arguments

None

### Returned Value

String      The URL string to an image.

### Example

```
Dim mRasterShieldSymbol  
Set mRasterShieldSymbol = Server.CreateObject("aims.RasterShieldSymbol")  
mRasterShieldSymbol.URL = "http://www.esri.com/Picture.gif"
```

## Recordset.BOF property

### Description

The Recordset.BOF property indicates the beginning of a recordset. BOF is True when the recordset is first retrieved. MoveFirst() method on a recordset will set BOF to False.

### Syntax

Recordset.BOF

### Arguments

None

### Returned Value

Boolean      Returns True when recordset is first retrieved and False otherwise.

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim mRecordset  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
Set mMap = Server.CreateObject( "aims.Map" )  
mResult = mMap.InitMap( mArcIMSConnector, "World" )  
Set mRecordset = mMap.Layers.Item(1).Recordset  
mRecordset.MoveLast()  
While Not mRecordset.BOF  
    mRecordset.MovePrevious()  
Wend
```

### See Also

Recordset.EOF

## **Recordset.CacheSize property**

### **Description**

The Recordset.CacheSize property returns or sets the number of records in a recordset that will be retrieved per a single request to the ArcIMS Spatial Server.

### **Syntax**

Recordset.CacheSize

### **Arguments**

None

### **Returned Value**

Long            Size of the cache (number of records). The default is 10.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mRecordset  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
Set mMap = Server.CreateObject( "aims.Map" )  
mResult = mMap.InitMap( mArcIMSConnector, "World" )  
Set mRecordset = mMap.Layers.Item(1).Recordset  
mRecordset.CacheSize = 15
```

## **Recordset.Clone method**

### **Description**

The Recordset.Clone method clones the recordset.

### **Syntax**

Recordset.Clone()

### **Arguments**

None

### **Returned Value**

Recordset      The clone of the recordset.

### **See Also**

FeatureLayer.Clone

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mRecordset  
Dim mClone  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
Set mMap = Server.CreateObject( "aims.Map" )  
mResult = mMap.InitMap( mArcIMSConnector, "World" )  
Set mRecordset = mMap.Layers.Item(1).Recordset  
Set mClone = mRecordset.Clone()
```

## **Recordset.Count property**

### **Description**

The Recordset.Count property returns the number of records in a recordset.

### **Syntax**

Recordset.Count

### **Arguments**

None

### **Returned Value**

Long            Number of records in the recordset.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mRecordset  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
Set mMap = Server.CreateObject( "aims.Map" )  
mResult = mMap.InitMap( mArcIMSConnector, "World" )  
Set mRecordset = mMap.Layers.Item(1).Recordset  
mCount = mRecordset.Count
```

## Recordset.Envelope property

### Description

The Recordset.Envelope property returns the Envelope for the current record in a recordset. The Envelope is the spatial extent of a record for a FeatureLayer feature. This property is not accessible until the recordset's MoveFirst or MoveLast methods have first been invoked.

### Syntax

Recordset.Envelope

### Arguments

None

### Returned Value

Envelope      Envelope of the current record in a recordset.

### Example

```
Dim mArcIMSConnector
Dim mMap
Dim mRecordset
Dim mEnvelope
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )
Set mMap = Server.CreateObject( "aims.Map" )
mResult = mMap.InitMap( mArcIMSConnector, "World" )
Set mRecordset = mMap.Layers.Item(1).Recordset
if mRecordset.MoveFirst() then
  Set mEnvelope = mRecordset.Envelope
end if
```

### See Also

Envelope

## Recordset.EOF property

### Description

The Recordset.EOF property indicates the end of the recordset. This property returns True when the current internal recordset pointer is pointing at the end of the recordset. A MoveNext() method on a recordset will cause an error if the recordset's EOF property is True.

### Syntax

Recordset.EOF

### Arguments

None

### Returned Value

Boolean      Returns True if last recordset is reached and False otherwise.

### See Also

Recordset.BOF

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim mRecordset  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
Set mMap = Server.CreateObject( "aims.Map" )  
mResult = mMap.InitMap( mArcIMSConnector, "World" )  
Set mRecordset = mMap.Layers.Item(1).Recordset  
While Not mRecordset.EOF  
    mRecordset.MoveNext()  
Wend
```

## Recordset.Fields property

### Description

The Recordset.Fields property returns a Fields object from the current recordset. The Fields object is not accessible until a MoveFirst() or MoveLast() method is invoked on the recordset.

### Syntax

Recordset.Fields

### Arguments

None

### Returned Value

Fields      Current Fields object.

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim mRecordset  
Dim mFields  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
Set mMap = Server.CreateObject( "aims.Map" )  
mResult = mMap.InitMap( mArcIMSConnector, "World" )  
Set mRecordset = mMap.Layers.Item(1).Recordset  
if mRecordset.MoveFirst() then  
    Set mFields = mRecordset.Fields  
end if
```

### See Also

Fields  
Recordset.MoveFirst  
Recordset.MoveNext  
Recordset.MovePrevious  
Recordset.MoveLast

### Note

Also refer to samples:  
[Query\\_Attribute\\_Data.asp](#)  
[Identify.asp](#)

## Recordset.Filter property

### Description

The Recordset.Filter property returns or sets the filter object for the recordset. The filter of a recordset from a FeatureLayer is independent of the FeatureLayer's filter object.

### Syntax

Recordset.Filter

### Arguments

None

### Returned Value

Filter      Filter object.

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim mFilter  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
Set mMap = Server.CreateObject("aims.Map")  
mResult = mMap.InitMap(mArcIMSConnector, "World")  
Set mFilter = mMap.Layers.Item(1).Recordset.Filter
```

### See Also

Filter  
FeatureLayer.Filter

### Note

Also refer to the following samples:

[Query\\_Attribute\\_Data.asp](#)  
[Identify.asp](#)

## Recordset.MoveFirst method

### Description

The Recordset.MoveFirst method sets the internal pointer to the first record in a recordset and returns a boolean value indicating the success of the method. If the recordset is empty, MoveFirst() will return FALSE.

### Syntax

Recordset.MoveFirst()

### Arguments

None

### Returned Value

Boolean      Returns True if pointer was set to the first record and False if there are 0 records in the recordset.

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim mRecordset  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
Set mMap = Server.CreateObject( "aims.Map" )  
mResult = mMap.InitMap( mArcIMSConnector, "World" )  
Set mRecordset = mMap.Layers.Item(1).Recordset  
mResult = mRecordset.MoveFirst()
```

### See Also

[Recordset.MoveNext](#)  
[Recordset.MoveLast](#)  
[Recordset.MovePrevious](#)

### Note

Also refer to samples:  
[Query\\_Attribute\\_Data.asp](#)  
[Identify.asp](#)

## **Recordset.MoveLast method**

### **Description**

The Recordset.MoveLast method sets the internal pointer to the last record in a recordset and returns a boolean value indicating the success of the method. If the recordset is empty, MoveLast() will return FALSE.

### **Syntax**

Recordset.MoveLast()

### **Arguments**

None

### **Returned Value**

Boolean      Returns True if pointer was set to the last record and False if there are 0 records in the recordset.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mRecordset  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
Set mMap = Server.CreateObject( "aims.Map" )  
mResult = mMap.InitMap( mArcIMSConnector, "World" )  
Set mRecordset = mMap.Layers.Item(1).Recordset  
mResult = mRecordset.MoveLast()
```

### **See Also**

[Recordset.MoveNext](#)  
[Recordset.MoveFirst](#)  
[Recordset.MovePrevious](#)

## Recordset.MoveNext method

### Description

The Recordset.MoveNext method sets the internal pointer to the next record in a recordset and returns a boolean value indicating the success of the method. If the recordset pointer is at the last record, MoveNext() will return False.

### Syntax

Recordset.MoveNext()

### Arguments

None

### Returned Value

Boolean      Returns True if pointer was set to the next record and False if recordset's EOF property is TRUE.

### Example

```
Dim mArcIMSConnector
Dim mMap
Dim mRecordset
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )
Set mMap = Server.CreateObject( "aims.Map" )
mResult = mMap.Init( mArcIMSConnector, "World" )
Set mRecordset = mMap.Layers.Item(1).Recordset
if mRecordset.MoveFirst() then
    Do
        ...
        Loop While mRecordset.MoveNext()
    end if
```

### See Also

[Recordset.MoveFirst](#)  
[Recordset.MoveLast](#)  
[Recordset.MovePrevious](#)

### Note

Also refer to the following samples:  
[Query\\_Attribute\\_Data.asp](#)

## Recordset.MovePrevious method

### Description

The Recordset.MovePrevious method sets the internal pointer to the previous record in a recordset and returns a boolean value indicating the success of the method. If the recordset pointer is at the first record, MovePrevious() will return False.

### Syntax

Recordset.MovePrevious()

### Arguments

None

### Returned Value

Boolean      Returns True if pointer was set to the previous record and False if recordset's BOF property is TRUE.

### Example

```
Dim mArcIMSConnector  
Dim mMap  
Dim mRecordset  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
Set mMap = Server.CreateObject( "aims.Map" )  
mResult = mMap.InitMap( mArcIMSConnector, "World" )  
Set mRecordset = mMap.Layers.Item(1).Recordset  
mResult = mRecordset.MoveLast()  
mResult = mRecordset.MovePrevious()
```

### See Also

[Recordset.MoveFirst](#)  
[Recordset.MoveLast](#)  
[Recordset.MoveNext](#)

## **Recordset.TableDesc property**

### **Description**

The Recordset.TableDesc property returns a TableDesc object that contains a description of the fields in a recordset.

### **Syntax**

Recordset.TableDesc

### **Arguments**

None

### **Returned Value**

TableDesc      TableDesc object which contains a description of the fields in a recordset.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mTableDesc  
Set mArcIMSConnector = Server.CreateObject( "aims.ArcIMSConnector" )  
Set mMap = Server.CreateObject( "aims.Map" )  
mResult = mMap.InitMap( mArcIMSConnector, "World" )  
Set mTableDesc = mMap.Layers.Item(1).Recordset.TableDesc
```

### **See Also**

TableDesc

### **Note**

Also refer to the following samples:

[Query\\_Attribute.asp](#)

## **ScaleBarObject.Antialiasing property**

### **Description**

The ScaleBarObject.Antialiasing property turns antialiasing on/off. Antialiasing allows lines to appear smooth by coloring adjacent pixels in such a way as to give the edge of a line a more gradual fade to the background.

### **Syntax**

ScaleBarObject.Antialiasing

### **Arguments**

None

### **Returned Value**

Boolean      True indicates that antialiasing is on. False indicates that it is off.

### **Example**

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## ScaleBarObject.BackColor property

### Description

The ScaleBarObject.BackColor property returns or sets a color constant indicating the background color of a ScaleBarObject.

### Syntax

ScaleBarObject.BackColor

### Arguments

None

### Returned Value

imsColor              Background color constant. The default is imsBlack (0).

### Example

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## ScaleBarObject.BarColor property

### Description

The ScaleBarObject.BarColor property returns or sets a color constant for the ScaleBarObject color.

### Syntax

ScaleBarObject.BarColor

### Arguments

None

### Returned Value

imsColor              ScaleBarObject color constant. The default value is 0.

### Example

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## ScaleBarObject.BarTransparency property

### Description

The ScaleBarObject.BarTransparency property returns or sets the ScaleBarObject's transparency coefficient. A low number indicates greater transparency for the ScaleBarObject.

### Syntax

ScaleBarObject.BarTransparency

### Arguments

None

### Returned Value

Double      ScaleBarObject transparency coefficient. The default value is 1.

### Example

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mScaleBar.BarTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## **ScaleBarObject.BarWidth property**

### **Description**

The ScaleBarObject.BarWidth property returns or sets the ScaleBarObject width in pixels.

### **Syntax**

ScaleBarObject.BarWidth

### **Arguments**

None

### **Returned Value**

Long Bar width. The default value is 0.

### **Example**

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## ScaleBarObject.Distance property

### Description

The ScaleBarObject.Distance property returns or sets the distance of the ScaleBarObject in map units.

### Syntax

ScaleBarObject.Distance

### Arguments

None

### Returned Value

Double      Distance. The default value is 0. This value is the calculated distance of the in setting map unit from the value of the Scalebar.ScreenLength.

### Example

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
mScaleBar.Distance = 100  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## ScaleBarObject.Font property

### Description

The ScaleBarObject.Font property returns or sets the font name string for the text labels used in the ScaleBarObject.

### Syntax

ScaleBarObject.Font

### Arguments

None

### Returned Value

Font            Font name. The default value is “Arial”.

### Example

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## ScaleBarObject.FontColor property

### Description

The ScaleBarObject.FontColor property returns or sets a color constant for the font of the text labels that are used for the ScaleBarObject.

### Syntax

ScaleBarObject.FontColor

### Arguments

None

### Returned Value

imsColor                  Font color constant. The default value is imsBlack (0).

### See Also

imsColor

### Example

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0
```

```
mScaleBar.BackColor = imsColor.imsWhite
```

```
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## **ScaleBarObject.FontSize property**

### **Description**

The ScaleBarObject.FontSize property returns or sets the font size for text labels used with the ScaleBarObject.

### **Syntax**

ScaleBarObject.FontSize

### **Arguments**

None

### **Returned Value**

Long            Font size. The default value is 0.

### **Example**

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mScaleBar.BarTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## ScaleBarObject.FontStyle property

### Description

The ScaleBarObject.FontStyle property returns or sets the font style constant for the text used with the ScaleBarObject. Possible values are:

- 0 = imsRegular
- 1 = imsBold
- 2 = imsItalic
- 3 = imsUnderline
- 4 = imsOutline

### Syntax

ScaleBarObject.FontStyle

### Arguments

None

### Returned Value

imsFontStyle     Font style constant. The default value is imsRegular (0).

### Example

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true
```

```
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mScaleBar.BarTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

#### See Also

[imsFontStyle](#)

## **ScaleBarObject.Id property**

### **Description**

The ScaleBarObject.Id property returns or sets a string identifier for a ScaleBarObject.

### **Syntax**

ScaleBarObject.Id

### **Arguments**

None

### **Returned Value**

String      ScaleBarObject's Id. The default value is "ScaleBar".

### **Example**

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mScaleBar.BarTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## ScaleBarObject.MapUnits property

### Description

The ScaleBarObject.MapUnits property returns or sets a constant that indicates the units of the ScaleBarObject. Available values are:

- 0 = imsDecimalDegrees
- 1 = imsMiles
- 2 = imsFeet
- 3 = imsKilometers
- 4 = imsMeters
- 5 = imsInches
- 6 = imsCentimeters

### Syntax

ScaleBarObject.MapUnits

### Arguments

None

### Returned Value

imsMapUnits      Scalebar units constant. The default value is imsDecimalDegrees (0).

### Example

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true
```

```
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mScaleBar.BarTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

#### **See Also**

[imsMapUnits](#)

## ScaleBarObject.Name property

### Description

The ScaleBarObject.Name property returns or sets the name of the ScaleBarObject.

### Syntax

ScaleBarObject.Name

### Arguments

None

### Returned Value

String      ScaleBarObject's name. The default value is "ScaleBar".

### Example

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mScaleBar.BarTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## ScaleBarObject.Overlap property

### Description

The ScaleBarObject.Overlap property returns or sets a Boolean value indicating whether or not labels should be allowed to overlap the ScaleBarObject.

### Syntax

```
ScaleBarObject.Overlap
```

### Arguments

None

### Returned Value

Boolean      True indicates that labeling can overlap the ScaleBarObject. False means that no labels will be allowed to overlap the ScaleBarObject. The default value is False.

### Example

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0
```

```
mScaleBar.BarTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## ScaleBarObject.Precision property

### Description

The ScaleBarObject.Precision property returns or sets the number of decimal places displayed for the ScaleBarObject units.

### Syntax

ScaleBarObject.Precision

### Arguments

None

### Returned Value

Long            Number of decimal places. The default value is 0.

### Example

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mScaleBar.BarTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add(mScaleBar, 0)
```

## ScaleBarObject.ScaleUnits property

### Description

The ScaleBarObject.ScaleUnits property returns or sets a constant indicating the display units of the ScaleBarObject. The available values are:

- 0 = imsDecimalDegrees
- 1 = imsMiles
- 2 = imsFeet
- 3 = imsKilometers
- 4 = imsMeters
- 5 = imsInches
- 6 = imsCentimeters

### Syntax

ScaleBarObject.ScaleUnits

### Arguments

None

### Returned Value

imsMapUnits      Display units constant. The default value is imsDecimalDegrees (0).

### See Also

imsMapUnits

### Example

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees
```

```
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mScaleBar.BarTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## ScaleBarObject.ScreenLength property

### Description

The ScaleBarObject.ScreenLength property returns or sets the ScaleBarObject length in screen pixels.

### Syntax

ScaleBarObject.ScreenLength

### Arguments

None

### Returned Value

Long Pixel length. The default value is 0.

### Example

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mScaleBar.BarTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## **ScaleBarObject.TextTransparency property**

### **Description**

The ScaleBarObject.TextTransparency property returns or sets the coefficient of the text transparency for the ScaleBarObject's text. The lower the number, the greater the transparency.

### **Syntax**

ScaleBarObject.TextTransparency

### **Arguments**

None

### **Returned Value**

Double      Text transparency. The default value is 1.0.

### **Example**

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mScaleBar.BarTransparency = 1.0  
mScaleBar.BackColor = imsColor.imsWhite  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## ScaleBarObject.X property

### Description

The ScaleBarObject.X property returns or sets the X coordinate of the ScaleBarObject location.

### Syntax

ScaleBarObject.X

### Arguments

None

### Returned Value

Double        X coordinate. The default value is 0.

### See Also

ScaleBarObject.Y

### Example

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.FontName = "Arial"  
mScaleBar.FontColor = imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = acBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## ScaleBarObject.Y property

### Description

The ScaleBarObject.Y property returns or sets the Y coordinate of the ScaleBarObject location.

### Syntax

ScaleBarObject.Y

### Arguments

None

### Returned Value

Double      Y coordinate. The default value is 0.

### See Also

ScaleBarObject.X

### Example

```
dim mScalebar  
set mScalebar = CreateObject("aims.ScaleBarObject")  
mScalebar.x = 25  
mScalebar.y = 25  
mScalebar.BarWidth = 10.0  
mScaleBar.Font = "Arial"  
mScaleBar.FontColor = imsColor.imsBlack  
mScalebar.FontSize = 12  
mScaleBarObject.FontStyle = imsFontStyle.imsRegular  
mScalebar.BarColor = imsColor.imsBlack  
mScalebar.MapUnits = 0 ' degrees  
mScalebar.ScaleUnits = 0 ' kilometers  
mScalebar.ScreenLength = 50  
mScalebar.Antialiasing = true  
mScaleBar.Overlap = False  
mScaleBar.Id = "ScaleBar:1"  
mScaleBar.Name = "MyScaleBar"  
mScaleBar.Precision = 2  
mScaleBar.TextTransparency = 1.0
```

```
mScaleBar.BackColor = imsColor.imsWhite  
  
mResult = Map.Layers.Add(AcetateLayer1)  
mResult = AcetateLayer1.Add( mScaleBar, 0)
```

## **ScaleDependentRenderer.Clone method**

### **Description**

The ScaleDependentRenderer.Clone method clones the ScaleDependentRenderer.

### **Syntax**

ScaleDependentRenderer.Clone()

### **Argument**

None

### **Returned Value**

ScaleDependentRenderer      Cloned renderer.

### **Example**

```
dim mClone
dim mScaleDependentRenderer
set mScaleDependentRenderer = Server.CreateObject("aims.ScaleDependentRenderer")
set mClone = mScaleDependentRenderer.Clone()
mClone.Lower = 0.1
mClone.Upper = 0.01
set mRenderer = Server.CreateObject("aims.SimpleRenderer")
mClone.Renderer = mRenderer
```

## **ScaleDependentRenderer.Lower property**

### **Description**

The ScaleDependentRenderer.Lower property returns or sets the lower relative scale threshold of a ScaleDependentRenderer.

### **Syntax**

ScaleDependentRenderer.Lower

### **Arguments**

None

### **Returned Value**

Double      Lower scale threshold.

### **Example**

```
dim mClone
dim mScaleDependentRenderer
set mScaleDependentRenderer = Server.CreateObject("aims.ScaleDependentRenderer")
set mClone = mScaleDependentRenderer.Clone()
mClone.Lower = 0.1
mClone.Upper = 0.01
set mRenderer = Server.CreateObject("aims.SimpleRenderer")
mClone.Renderer = mRenderer
```

### **See Also**

[ScaleDependentRenderer.Upper](#)

## **ScaleDependentRenderer.Renderer** property

### **Description**

The ScaleDependentRenderer.Renderer property returns or sets the Renderer object that is to be displayed by the ScaleDependentRenderer. The Renderer will be under the constraints of the ScaleDependentRenderer properties.

### **Syntax**

ScaleDependentRenderer.Renderer

### **Arguments**

None

### **Returned Value**

Object                  Any renderer.

### **Example**

```
dim mClone
dim mScaleDependentRenderer
set mScaleDependentRenderer = Server.CreateObject("aims.ScaleDependentRenderer")
set mClone = mScaleDependentRenderer.Clone()
mClone.Lower = 0.1
mClone.Upper = 0.01
set mRenderer = Server.CreateObject("aims.SimpleRenderer")
mClone.Renderer = mRenderer
```

## **ScaleDependentRenderer.Upper property**

### **Description**

The ScaleDependentRenderer.Upper property returns or sets the upper relative scale threshold for the ScaleDependentRenderer.

### **Syntax**

ScaleDependentRenderer.Upper

### **Arguments**

None

### **Returned Value**

Double      Upper scale threshold. The default value is 0.

### **Example**

```
dim mClone
dim mScaleDependentRenderer
set mScaleDependentRenderer = Server.CreateObject("aims.ScaleDependentRenderer")
set mClone = mScaleDependentRenderer.Clone()
mClone.Lower = 0.1
mClone.Upper = 0.01
set mRenderer = Server.CreateObject("aims.SimpleRenderer")
mClone.Renderer = mRenderer
```

### **See Also**

[ScaleDependentRenderer.Lower](#)

## SDEWorkspace.DataBase property

### Description

The SDEWorkspace.DataBase property returns or sets the ArcSDE database name for an SDEWorkspace.

### Syntax

SDEWorkspace.DataBase

### Arguments

None

### Returned Value

String        ArcSDE database name.

### Example

```
dim mSDEWorkspace
dim lay
dim mConn
dim mMap
set mConn = server.CreateObject("aims.ArcIMSConnector")
mConn.ServerName = "husker"
mConn.ServerPort = 5300
set mMap = server.CreateObject("aims.Map")
mMap.InitMap mConn,"World"
Set mSDEWorkspace = Server.CreateObject("aims.SDEWorkspace")
MSDEWorkspace.DataBase = "SDE1"
mSDEWorkspace.User = "sde"
mSDEWorkspace.Password = "sde"
mSDEWorkspace.Server = "brie"
mSDEWorkspace.Instance = "brie_sde"
mSDEWorkspace.name = "SDE_AWS_CONT"
mSDEWorkspace.FeatureClass = imsFeatureClass.imsPolygon
set Lay = mMap.Layers.Create(mSDEWorkSpace)
mMap.Refresh
```

## SDEWorkspace.FeatureClass property

### Description

The SDEWorkspace.FeatureClass property returns or sets the FeatureClass type constant of the layer to be specified in the SDEWorkspace. The possible values are:

- 1 = imsPoint
- 2 = imsLine
- 3 = imsPolygon

### Syntax

SDEWorkspace.FeatureClass

### Arguments

None

### Returned Value

imsFeatureClass      FeatureClass type constant of the layer to be specified in the SDEWorkspace. The default value is 0.

### Example

```
dim mSDEWorkspace
dim lay
dim mConn
dim mMap
set mConn = server.CreateObject("aims.ArcIMSConnector")
mConn.ServerName = "husker"
mConn.ServerPort = 5300
set mMap = server.CreateObject("aims.Map")
mMap.InitMap mConn,"World"
Set mSDEWorkspace = Server.CreateObject("aims.SDEWorkspace")
mSDEWorkspace.User = "mdw"
mSDEWorkspace.Password = "mdw"
mSDEWorkspace.Server = "brie"
mSDEWorkspace.FeatureClass = imsFeatureClass.imsPoint
mSDEWorkspace.Name = "mdw.facil.gid"

set lay = mMap.Layers.Create(mSDEWorkspace)
```

## SDEWorkspace.Geoindexer property

### Description

The SDEWorkspace.GeoIndexer property returns or sets the name of the directory where geocoding index will be built for the SDEWorkspace layer.

### Syntax

SDEWorkspace.Geoindexer

### Arguments

None

### Returned Value

String      Directory path where the geocoding index will be built.

### Example

```
dim mSDEWorkspace  
dim lay  
dim mConn  
dim mMap  
set mConn = server.CreateObject("aims.ArcIMSConnector")  
mConn.ServerName = "husker"  
mConn.ServerPort = 5300  
set mMap = server.CreateObject("aims.Map")  
mMap.InitMap mConn,"World"  
Set mSDEWorkspace = Server.CreateObject("aims.SDEWorkspace")  
mSDEWorkspace.User = "mwd"  
mSDEWorkspace.Password = "mwd"  
mSDEWorkspace.Server = "brie"  
mSDEWorkspace.FeatureClass = imsFeatureClass.imsLine  
mSDEWorkspace.Name = "mwd.streets.gid"  
mSDEWorkspace.Geoindexer = "C:\Temp\Indexer"
```

## SDEWorkspace.Id property

### Description

The SDEWorkspace.Id property returns or sets a string identifier for the SDEWorkspace.

### Syntax

SDEWorkspace.Id

### Arguments

None

### Returned Value

String            SDEWorkspace's Id. The default value begins with "sde\_ws\_" followed by a random number, for example, "sde\_ws-29816".

### Example

```
dim mSDEWorkspace
dim lay
dim mConn
dim mMap
set mConn = server.CreateObject("aims.ArcIMSConnector")
mConn.ServerName = "husker"
mConn.ServerPort = 5300
set mMap = server.CreateObject("aims.Map")
mMap.InitMap mConn,"World"
Set mSDEWorkspace = Server.CreateObject("aims.SDEWorkspace")
mSDEWorkspace.User = "mdw"
mSDEWorkspace.Password = "mdw"
mSDEWorkspace.Server = "brie"
mSDEWorkspace.FeatureClass = imsFeatureClass.imsLine
mSDEWorkspace.Name = "mdw.streets.gid"
mSDEWorkspace.Id = "SDEWorkspace:3"
```

## SDEWorkspace.Instance property

### Description

The SDEWorkspace.Instance property returns or sets a string indicating an ArcSDE instance name for the SDEWorkspace.

### Syntax

SDEWorkspace.Instance

### Arguments

None

### Returned Value

String            ArcSDE instance name.

### Example

```
dim mSDEWorkspace
dim lay
dim mConn
dim mMap
set mConn = server.CreateObject("aims.ArcIMSConnector")
mConn.ServerName = "husker"
mConn.ServerPort = 5300
set mMap = server.CreateObject("aims.Map")
mMap.InitMap mConn,"World"
Set mSDEWorkspace = Server.CreateObject("aims.SDEWorkspace")
mSDEWorkspace.User = "mwd"
mSDEWorkspace.Password = "mwd"
mSDEWorkspace.Server = "brie"
mSDEWorkspace.FeatureClass = imsFeatureClass.imsLine
mSDEWorkspace.Name = "mwd.streets.gid"
mSDEWorkspace.Instance = "brie_sde"

set Lay = mMap.Layers.Create(mSDEWorkSpace)
mMap.Refresh
```

## SDEWorkspace.Name property

### Description

The SDEWorkspace.Name property returns or sets the name of a layer to be specified in an SDEWorkspace.

### Syntax

SDEWorkspace.Name

### Arguments

None

### Returned Value

String           Layer's name.

### Example

```
dim mSDEWorkspace  
dim lay  
dim mConn  
dim mMap  
set mConn = server.CreateObject("aims.ArcIMSConnector")  
mConn.ServerName = "husker"  
mConn.ServerPort = 5300  
set mMap = server.CreateObject("aims.Map")  
mMap.InitMap mConn,"World"  
Set mSDEWorkspace = Server.CreateObject("aims.SDEWorkspace")  
mSDEWorkspace.User = "mdw"  
mSDEWorkspace.Password = "mdw"  
mSDEWorkspace.Server = "brie"  
mSDEWorkspace.FeatureClass = imsFeatureClass.imsLine  
mSDEWorkspace.Name = "Country"
```

## SDEWorkspace.Password property

### Description

The SDEWorkspace.Password property returns or sets the ArcSDE password string of a user in an SDEWorkspace.

### Syntax

SDEWorkspace.Password

### Arguments

None

### Returned Value

String            ArcSDE password for user.

### Example

```
dim mSDEWorkspace
dim lay
dim mConn
dim mMap
set mConn = server.CreateObject("aims.ArcIMSConnector")
mConn.ServerName = "husker"
mConn.ServerPort = 5300
set mMap = server.CreateObject("aims.Map")
mMap.InitMap mConn,"World"
Set mSDEWorkspace = Server.CreateObject("aims.SDEWorkspace")
mSDEWorkspace.User = "mwd"
mSDEWorkspace.Password = "mwd"
mSDEWorkspace.Server = "brie"
mSDEWorkspace.FeatureClass = imsFeatureClass.imsLine
mSDEWorkspace.Name = "Country"

set Lay = mMap.Layers.Create(mSDEWorkSpace)
mMap.Refresh
```

## SDEWorkspace.Server property

### Description

The SDEWorkspace.Server property returns or sets the name of an ArcSDE server that the SDEWorkspace object will reference.

### Syntax

SDEWorkspace.Server

### Arguments

None

### Returned Value

String      ArcSDE server name.

### Example

```
dim mSDEWorkspace
dim lay
dim mConn
dim mMap
set mConn = server.CreateObject("aims.ArcIMSConnector")
mConn.ServerName = "husker"
mConn.ServerPort = 5300
set mMap = server.CreateObject("aims.Map")
mMap.InitMap mConn,"World"
Set mSDEWorkspace = Server.CreateObject("aims.SDEWorkspace")
mSDEWorkspace.User = "mdw"
mSDEWorkspace.Password = "mdw"
mSDEWorkspace.Server = "brie"
mSDEWorkspace.FeatureClass = imsFeatureClass.imsLine
mSDEWorkspace.Name = "Country"

set Lay = mMap.Layers.Create(mSDEWorkSpace)
mMap.Refresh
```

## SDEWorkspace.User property

### Description

The SDEWorkspace.User property returns or sets the ArcSDE user name for an SDEWorkspace object.

### Syntax

SDEWorkspace.User

### Arguments

None

### Returned Value

String            ArcSDE user name.

### Example

```
dim mSDEWorkspace
dim lay
dim mConn
dim mMap
set mConn = server.CreateObject("aims.ArcIMSConnector")
mConn.ServerName = "husker"
mConn.ServerPort = 5300
set mMap = server.CreateObject("aims.Map")
mMap.InitMap mConn,"World"
Set mSDEWorkspace = Server.CreateObject("aims.SDEWorkspace")
mSDEWorkspace.User = "mwd"
mSDEWorkspace.Password = "mwd"
mSDEWorkspace.Server = "brie"
mSDEWorkspace.FeatureClass = imsFeatureClass.imsLine
mSDEWorkspace.Name = "Country"

set Lay = mMap.Layers.Create(mSDEWorkSpace)
mMap.Refresh
```

## **ShapeWorkspace.Directory property**

### **Description**

The ShapeWorkspace.Directory property returns or sets the pathname of a directory containing shapefile data.

### **Syntax**

ShapeWorkspace.Directory

### **Arguments**

None

### **Returned Value**

String      Directory containing shapefile data.

### **Example**

```
dim mShapeWorkspace  
dim lay  
dim mConn  
dim mMap  
set mConn = server.CreateObject("aims.ArcIMSConnector")  
mConn.ServerName = "husker"  
mConn.ServerPort = 5300  
set mMap = server.CreateObject("aims.Map")  
set mShapeWorkspace = Server.CreateObject("aims.ShapeWorkspace")  
mShapeWorkspace.Directory = "D:\EsriData\Shapes"  
mShapeWorkspace.FeatureClass = imsFeatureClass.imsPolygon  
mShapeWorkspace.Name = "Countries"  
set Lay = mMap.Layers.Create(mShapeWorkSpace)  
mMap.Refresh
```

## **ShapeWorkspace.FeatureClass property**

### **Description**

The ShapeWorkspace.FeatureClass property returns or sets the type of feature class for a layer to be specified in a ShapeWorkspace. The possible values are:

- 1 = imsPoint
- 2 = imsLine
- 3 = imsPolygon

### **Syntax**

ShapeWorkspace.FeatureClass

### **Arguments**

None

### **Returned Value**

imsFeatureClass      FeatureClass of the layer being specified.

### **Example**

```
dim mShapeWorkspace
dim lay
dim mConn
dim mMap
set mConn = server.CreateObject("aims.ArcIMSConnector")
mConn.ServerName = "husker"
mConn.ServerPort = 5300
set mMap = server.CreateObject("aims.Map")
set mShapeWorkspace = Server.CreateObject("aims.ShapeWorkspace")
mShapeWorkspace.Directory = "D:\EsriData\Shapes"
mShapeWorkspace.FeatureClass = imsFeatureClass.imsPolygon
mShapeWorkspace.Name = "Countries"
set Lay = mMap.Layers.Create(mShapeWorkSpace)
mMap.Refresh
```

### **See Also**

imsFeatureClass

## **ShapeWorkspace.Id property**

### **Description**

The ShapeWorkspace.Id property returns or sets a string identifier for a ShapeWorkspace.

### **Syntax**

ShapeWorkspace.Id

### **Arguments**

None

### **Returned Value**

String      The identifier of the ShapeWorkspace. The default value begins with “shp\_ws-“ and is followed by a random number, for example, “shp\_ws-76072”.

### **Example**

```
dim mShapeWorkspace
dim lay
dim mConn
dim mMap
set mConn = server.CreateObject("aims.ArcIMSConnector")
mConn.ServerName = "husker"
mConn.ServerPort = 5300
set mMap = server.CreateObject("aims.Map")
set mShapeWorkspace = Server.CreateObject("aims.ShapeWorkspace")
mShapeWorkspace.Directory = "D:\EsriData\Shapes"
mShapeWorkspace.FeatureClass = imsFeatureClass.imsPolygon
mShapeWorkspace.Name = "Countries"
mShapeWorkspace.Id = "ID1"
set Lay = mMap.Layers.Create(mShapeWorkSpace)
mMap.Refresh
```

## **ShapeWorkspace.Name property**

### **Description**

The ShapeWorkspace.Name property returns or sets the name of the layer to be specified in the ShapeWorkspace.

### **Syntax**

ShapeWorkspace.Name

### **Arguments**

None

### **Returned Value**

String            Name of the layer.

### **Example**

```
dim mShapeWorkspace  
dim lay  
dim mConn  
dim mMap  
  
set mConn = server.CreateObject("aims.ArcIMSConnector")  
mConn.ServerName = "husker"  
mConn.ServerPort = 5300  
  
set mMap = server.CreateObject("aims.Map")  
set mShapeWorkspace = Server.CreateObject("aims.ShapeWorkspace")  
mShapeWorkspace.Directory = "D:\EsriData\Shapes"  
mShapeWorkspace.FeatureClass = imsFeatureClass.imsPolygon  
mShapeWorkspace.Name = "Countries"  
set Lay = mMap.Layers.Create(mShapeWorkSpace)  
  
mMap.Refresh
```

## **ShieldSymbol.Clone method**

### **Description**

The ShieldSymbol.Clone method clones the ShieldSymbol.

### **Syntax**

ShieldSymbol.Clone ()

### **Arguments**

None

### **Returned Value**

ShieldSymbol Cloned symbol.

### **Example**

```
dim mClone  
dim mShieldSymbol  
Set mShieldSymbol = Server.CreateObject("aims.ShieldSymbol")  
Set mClone = mShieldSymbol.Clone()
```

## **ShieldSymbol.Font property**

### **Description**

The ShieldSymbol.Font property returns or sets the name of the font used with the ShieldSymbol.

### **Syntax**

ShieldSymbol.Font

### **Arguments**

None

### **Returned Value**

String            Font name. The default value is “default”.

### **Example**

```
Dim mShieldSymbol  
Set mShieldSymbol = Server.CreateObject("aims.ShieldSymbol")  
mShieldSymbol.Font = "Arial"
```

## **ShieldSymbol.FontColor property**

### **Description**

The ShieldSymbol.FontColor property returns or sets a font color constant for the font used with a ShieldSymbol.

### **Syntax**

ShieldSymbol.Color

### **Arguments**

None

### **Returned Value**

imsColor                  Font color. The default value is imsBlack (0).

### **Example**

```
Dim mShieldSymbol  
Set mShieldSymbol = Server.CreateObject("aims.ShieldSymbol")  
mShieldSymbol.Color = imsColor.imsRed
```

## **ShieldSymbol.FontSize property**

### **Description**

The ShieldSymbol.FontSize property returns or sets the font size for the font used with a ShieldSymbol.

### **Syntax**

ShieldSymbol.FontSize

### **Arguments**

None

### **Returned Value**

Long            Font size. The default value is 12.

### **Example**

```
Dim mShieldSymbol  
Set mShieldSymbol = Server.CreateObject("aims.ShieldSymbol")  
mShieldSymbol.FontSize = 10
```

## **ShieldSymbol.LabelMode property**

### **Description**

The ShieldSymbol.LabelMode property returns or sets a LabelMode constant that determines what labels are drawn in the ShieldSymbol. The possible values are:

0 = imsFull

1 = imsNumericonly

‘Full’ means that the entire value of the label field will be displayed in the ShieldSymbol. ‘Numericonly’ means that only the numeric parts of the label field value will be displayed.

### **Syntax**

ShieldSymbol.LabelMode

### **Arguments**

None

### **Returned Value**

imsLabelMode    LabelMode constant. The default value is ‘imsFull’ (0).

### **Example**

```
Dim mShieldSymbol  
Set mShieldSymbol = Server.CreateObject("aims.ShieldSymbol")  
mShieldSymbol.LabelMode = imsLabelMode.imsFull
```

### **See Also**

imsLabelMode

## **ShieldSymbol.MinSize property**

### **Description**

The ShieldSymbol.MinSize property returns or sets the minimum ShieldSymbol size. The size determines the minimum size in characters. By default, shields expand to the length of the text.

### **Syntax**

ShieldSymbol.MinSize

### **Arguments**

None

### **Returned Value**

Long            Minimum shield size in characters.

### **Example**

```
Dim mShieldSymbol  
Set mShieldSymbol = Server.CreateObject("aims.ShieldSymbol")  
mShieldSymbol.MinSize = 10
```

## **ShieldSymbol.Shadow property**

### **Description**

The ShieldSymbol.Shadow property returns or sets a color constant for the shadow color of a ShieldSymbol.

### **Syntax**

ShieldSymbol.Shadow

### **Arguments**

None

### **Returned Value**

imsColor      Shadow color constant. The default value is imsBlack (0).

### **Example**

```
Dim mShieldSymbol  
Set mShieldSymbol = Server.CreateObject("aims.ShieldSymbol")  
mShieldSymbol.Shadow = imsColor.imsBlack
```

## **ShieldSymbol.ShieldType** property

### **Description**

The ShieldSymbol.ShieldType property returns or sets a shield type constant for a ShieldSymbol. The possible values are:

- 0 = imsInterstate
- 1 = imsUSRoad
- 2 = imsRect
- 3 = imsOval
- 4 = imsMexican

### **Syntax**

`ShieldSymbol.ShieldType`

### **Arguments**

None

### **Returned Value**

`imsShieldType`   Symbol type constant. The default value is `imsInterstate` (0).

### **Example**

```
Dim mShieldSymbol  
Set mShieldSymbol = Server.CreateObject("aims.ShieldSymbol")  
mShieldSymbol.ShieldType = imsShieldType.imsRect
```

### **See Also**

`imsShieldType`

## **SimpleLabelRenderer.Clone method**

### **Description**

The SimpleLabelRenderer.Clone method clones the SimpleLabelRenderer.

### **Syntax**

SimpleLabelRenderer.Clone()

### **Arguments**

None

### **Returned Value**

SimpleLabelRenderer      Cloned renderer.

### **Example**

```
dim mClone  
dim mSimpleLabelRenderer  
set mSimpleLabelRenderer = Server.CreateObject("aims.SimpleLabelRenderer")  
set mClone = mSimpleLabelRenderer.Clone()  
mSimpleLabelRenderer.Field = "CNTRY_NAME"  
  
mSimpleLabelRenderer.FWeight = imsWeight.imsNo_weight  
mSimpleLabelRenderer.HMLabels = imsHMLabels.imsOne_label_per_name  
mSimpleLabelRenderer.LabelPriorities = "1,1,1,1,2,2,2"  
mSimpleLabelRenderer.LLPosition = imsLLPosition.imsPlaceNone  
  
mSimpleLabelRenderer.RotationalAngles = 30  
mSimpleLabelRenderer.Symbol = mTextSymbol  
mMap.Refresh()
```

## **SimpleLabelRenderer.Field property**

### **Description**

The SimpleLabelRenderer.Field property returns or sets the field name that contains the values for labeling features.

### **Syntax**

SimpleLabelRenderer.Field

### **Arguments**

None

### **Returned Value**

String      Field name containing text for labeling a feature.

### **Example**

```
dim mClone
dim mSimpleLabelRenderer
set mSimpleLabelRenderer = Server.CreateObject("aims.SimpleLabelRenderer")
set mClone = mSimpleLabelRenderer.Clone()
mSimpleLabelRenderer.Field = "CNTRY_NAME"

mSimpleLabelRenderer.FWeight = imsWeight.imsNo_weight
mSimpleLabelRenderer.HMLLabels = imsHMLLabels.imsOne_label_per_name
mSimpleLabelRenderer.LabelPriorities = "1,1,1,1,2,2,2,2"
mSimpleLabelRenderer.LLPosition = imsLLPosition.imsPlaceNone

mSimpleLabelRenderer.RotationalAngles = 30
mSimpleLabelRenderer.Symbol = mTextSymbol
mMap.Refresh()
```

## **SimpleLabelRenderer.FWeight property**

### **Description**

The SimpleLabelRenderer.FWeight property returns or sets a constant indicating the importance of Feature Weight during labeling. The possible values are:

- 0 = imsNo\_weight
- 1 = imsMed\_weight
- 2 = imsHigh\_weight

### **Syntax**

SimpleLabelRenderer.FWeight

### **Arguments**

None

### **Returned Value**

imsWeight      Feature weight. The default value is imsNo\_weight.

### **Example**

```
dim mClone
dim mSimpleLabelRenderer
set mSimpleLabelRenderer = Server.CreateObject("aims.SimpleLabelRenderer")
set mClone = mSimpleLabelRenderer.Clone()
mSimpleLabelRenderer.Field = "CNTRY_NAME"

mSimpleLabelRenderer.FWeight = imsWeight.imsNo_weight
mSimpleLabelRenderer.HMLabels = imsHMLabels.imsOne_label_per_name
mSimpleLabelRenderer.LabelPriorities = "1,1,1,1,2,2,2"
mSimpleLabelRenderer.LLPosition = imsLLPosition.imsPlaceNone

mSimpleLabelRenderer.RotationalAngles = 30
mSimpleLabelRenderer.Symbol = mTextSymbol
mMap.Refresh()
```

## **SimpleLabelRenderer.HTMLLabels property**

### **Description**

The SimpleLabelRenderer.HTMLLabels property returns or sets a constant indicating how many labels are to be drawn while labeling a feature. The possible values are:

- 0 = imsOne\_label\_per\_name
- 1 = imsOne\_label\_per\_shape
- 2 = imsOne\_label\_per\_part

### **Syntax**

SimpleLabelRenderer.HTMLLabels

### **Arguments**

None

### **Returned Value**

imsHTMLLabels    Constant (long) determining how many labels are to be drawn to label a feature.  
The default value is imsOne\_label\_per\_name (0).

### **Example**

```
dim mClone
dim mSimpleLabelRenderer
set mSimpleLabelRenderer = Server.CreateObject("aims.SimpleLabelRenderer")
set mClone = mSimpleLabelRenderer.Clone()
mSimpleLabelRenderer.Field = "CNTRY_NAME"

mSimpleLabelRenderer.FWeight = imsWeight.imsNo_weight
mSimpleLabelRenderer.HTMLLabels = imsHTMLLabels.imsOne_label_per_name
mSimpleLabelRenderer.LabelPriorities = "1,1,1,1,2,2,2,2"
mSimpleLabelRenderer.LLPosition = imsLLPosition.imsPlaceNone
mSimpleLabelRenderer.RotationalAngles = 30
mSimpleLabelRenderer.Symbol = mTextSymbol
mMap.Refresh()
```

### **See Also**

imsHTMLLabels

## **SimpleLabelRenderer.LabelPriorities property**

### **Description**

The SimpleLabelRenderer.LabelPriorities property returns or sets a string that determines where to place a label around a point. There are 8 positions around a point. The LabelPriorities string is a comma-separated list of ‘priorities’ for the label location at each of the eight positions: 1 being the highest, 8 being the lowest. For example, “1,1,0,0,2,3,4,4” means the following: try to place the label at locations 1 and 2. Do not place the label at locations 3 or 4. If the label can’t be placed at 1 or 2, try locations 5, then 6, then 7, and 8.

### **Syntax**

SimpleLabelRenderer.LabelPriorities

### **Arguments**

None

### **Returned Value**

String	String determining where to place the label around the point. The default string is “2,2,1,4,5,3,2,4”.
--------	--

### **Example**

```
dim mClone
dim mSimpleLabelRenderer
set mSimpleLabelRenderer = Server.CreateObject("aims.SimpleLabelRenderer")
set mClone = mSimpleLabelRenderer.Clone()
mSimpleLabelRenderer.Field = "CNTRY_NAME"

mSimpleLabelRenderer.FWeight = imsWeight.imsNo_weight
mSimpleLabelRenderer.HMLabels = imsHMLabels.imsOne_label_per_name
mSimpleLabelRenderer.LabelPriorities = "1,1,1,1,2,2,2,2"
mSimpleLabelRenderer.LLPosition = imsLLPosition.imsPlaceNone

mSimpleLabelRenderer.RotationalAngles = 30
mSimpleLabelRenderer.Symbol = mTextSymbol

mMap.Refresh()
```

## **SimpleLabelRenderer.LLPosition property**

### **Description**

The SimpleLabelRenderer.LLPosition property returns or sets a constant determining where to place a label on a line.

### **Syntax**

SimpleLabelRenderer.LLPosition

### **Arguments**

None

### **Returned Value**

imsLLPosition    Determines where on the line to place a label. The default value is imsPlaceAbove.

### **Example**

```
dim mClone
dim mSimpleLabelRenderer
set mSimpleLabelRenderer = Server.CreateObject("aims.SimpleLabelRenderer")
set mClone = mSimpleLabelRenderer.Clone()
mSimpleLabelRenderer.Field = "CNTRY_NAME"

mSimpleLabelRenderer.FWeight = imsWeight.imsNo_weight
mSimpleLabelRenderer.HMLabels = imsHMLabels.imsOne_label_per_name
mSimpleLabelRenderer.LabelPriorities = "1,1,1,1,2,2,2"
mSimpleLabelRenderer.LLPosition = imsLLPosition.imsPlaceNone

mSimpleLabelRenderer.RotationalAngles = 30
mSimpleLabelRenderer.Symbol = mTextSymbol
mMap.Refresh()
```

### **See Also**

imsLLPosition

## **SimpleLabelRenderer.LWeight property**

### **Description**

The SimpleLabelRenderer.LWeight property returns or sets a constant that determines the use of label weights to prioritize labeling importance. The possible values are:

- 0 = imsNo\_weight
- 1 = imsMed\_weight
- 2 = imsHigh\_weight

### **Syntax**

SimpleLabelRenderer.LWeight

### **Arguments**

None

### **Returned Value**

imsWeight      Label weight. The default value is imsNo\_weight.

### **Example**

```
Dim mSimpleLabelRenderer  
Set mSimpleLabelRenderer = Server.CreateObject("aims.SimpleLabelRenderer")  
mSimpleLabelRenderer.LWeight = imsWeight.imsNo_weight
```

### **See Also**

imsWeight

## **SimpleLabelRenderer.RotationalAngles property**

### **Description**

The SimpleLabelRenderer.RotationAngles property returns or sets the angle at which labels will be placed relative to the feature.

### **Syntax**

```
mSimpleLabelRenderer.RotationalAngles
```

### **Arguments**

None

### **Returned Value**

Long            The rotational angles. The default value is 0.

### **Example**

```
dim mClone  
dim mSimpleLabelRenderer  
set mSimpleLabelRenderer = Server.CreateObject("aims.SimpleLabelRenderer")  
set mClone = mSimpleLabelRenderer.Clone()  
mSimpleLabelRenderer.Field = "CNTRY_NAME"  
  
mSimpleLabelRenderer.FWeight = imsWeight.imsNo_weight  
mSimpleLabelRenderer.HTMLLabels = imsHTMLLabels.imsOne_label_per_name  
mSimpleLabelRenderer.LabelPriorities = "1,1,1,1,2,2,2,2"  
mSimpleLabelRenderer.LLPosition = imsLLPosition.imsPlaceNone  
  
mSimpleLabelRenderer.RotationalAngles = 30  
mSimpleLabelRenderer.Symbol = mTextSymbol  
mMap.Refresh()
```

## **SimpleLabelRenderer.Symbol** property

### **Description**

The SimpleLabelRenderer.Symbol property returns or sets a symbol object for a SimpleLabelRenderer. The symbol can be a ShieldSymbol, TextSymbol, CalloutMarkerSymbol, or RasterShieldSymbol.

### **Syntax**

SimpleLabelRenderer.Symbol

### **Arguments**

None

### **Returned Value**

TextSymbol      Symbol for renderer.

### **Example**

```
dim mClone
dim mSimpleLabelRenderer
set mSimpleLabelRenderer = Server.CreateObject("aims.SimpleLabelRenderer")
set mClone = mSimpleLabelRenderer.Clone()
mSimpleLabelRenderer.Field = "CNTRY_NAME"

mSimpleLabelRenderer.FWeight = imsWeight.imsNo_weight
mSimpleLabelRenderer.HMLabels = imsHMLabels.imsOne_label_per_name
mSimpleLabelRenderer.LabelPriorities = "1,1,1,1,2,2,2"
mSimpleLabelRenderer.LLPosition = imsLLPosition.imsPlaceNone

mSimpleLabelRenderer.RotationalAngles = 30
mSimpleLabelRenderer.Symbol = mTextSymbol
mMap.Refresh()
```

### **See Also**

[ShieldSymbol](#)  
[TextSymbol](#)  
[CalloutMarkerSymbol](#)  
[RasterShieldSymbol](#)

## **SimpleLineSymbol.Antialiasing property**

### **Description**

The SimpleLineSymbol.Antialiasing property returns or sets a Boolean value indicating whether or not antialiasing will be used while rendering the SimpleLineSymbol. Antialiasing allows lines to appear smooth by coloring adjacent pixels in such a way as to give the edge of a line a more gradual fade to the background.

### **Syntax**

SimpleLineSymbol.Antialiasing

### **Arguments**

None

### **Returned Value**

Boolean      Turns antialiasing on/off. The default value is False.

### **Example**

```
Dim mSimpleLineSymbol  
Set mSimpleLineSymbol = Server.CreateObject("aims.SimpleLineSymbol")  
mSimpleLineSymbol.Antialiasing = True
```

## SimpleLineSymbol.CapStyle property

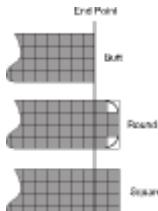
### Description

The SimpleLineSymbol.CapStyle property returns or sets a CapStyle constant that determines the line end style. The possible values are:

0 = imsRound

1 = imsButt

2 = imsSquare



### Syntax

SimpleLineSymbol.CapStyle

### Arguments

None

### Returned Value

imsCapStyle     Line end style constant. The default value is imsRound (0).

### Example

```
Dim mSimpleLineSymbol  
Set mSimpleLineSymbol = Server.CreateObject("aims.SimpleLineSymbol")  
mSimpleLineSymbol.CapStyle = imsCapStyle.imsRound
```

### See Also

imsCapStyle

## **SimpleLineSymbol.Clone method**

### **Description**

The SimpleLineSymbol.Clone method clones the SimpleLineSymbol.

### **Syntax**

SimpleLineSymbol.Clone ()

### **Arguments**

None

### **Returned Value**

SimpleLineSymbol      Cloned symbol.

### **Example**

```
dim mClone  
dim mSimpleLineSymbol  
Set mSimpleLineSymbol = Server.CreateObject("aims.SimpleLineSymbol")  
Set mClone = mSimpleLineSymbol.Clone()
```

## **SimpleLineSymbol.Color property**

### **Description**

The SimpleLineSymbol.Color property returns or sets a symbol color constant for the SimpleLineSymbol.

### **Syntax**

SimpleLineSymbol.Color

### **Arguments**

None

### **Returned Value**

imsColor      Symbol color constant. The default value is imsBlack (0).

### **Example**

```
Dim mSimpleLineSymbol  
Set mSimpleLineSymbol = Server.CreateObject("aims.SimpleLineSymbol")  
mSimpleLineSymbol.Color = imsColor.imsRed
```

### **See Also**

imsColor

## SimpleLineSymbol.JoinStyle property

### Description

The SimpleLineSymbol.JoinStyle property returns or sets a join style constant that determines how intersecting (or joined) lines are displayed with SimpleLineSymbols. The possible values are:

0 = imsRound



1 = imsMiter

2 = imsBevel

### Syntax

SimpleLineSymbol.JoinStyle

### Arguments

None

### Returned Value

imsJoinStyle     Line join style constant. The default value is imsRound (0).

### Example

```
Dim mSimpleLineSymbol  
Set mSimpleLineSymbol = Server.CreateObject("aims.SimpleLineSymbol")  
mSimpleLineSymbol.JoinStyle = imsJoinStyle.imsRound
```

### See Also

imsJoinStyle

## **SimpleLineSymbol.Overlap property**

### **Description**

The SimpleLineSymbol.Overlap property returns or sets a Boolean value indicating whether or not labels will overlap a SimpleLineSymbol.

### **Syntax**

SimpleLineSymbol.Overlap

### **Arguments**

None

### **Returned Value**

Boolean      True indicates that labels will be allowed to overlap the SimpleLineSymbol. False indicates that labels will not overlap the symbol. The default value is True.

### **Example**

```
Dim mSimpleLineSymbol  
Set mSimpleLineSymbol = Server.CreateObject("aims.SimpleLineSymbol")  
mSimpleLineSymbol.Overlap = False
```

## **SimpleLineSymbol.Style** property

### **Description**

The SimpleLineSymbol.Style property returns or sets a constant that determines a line style for a SimpleLineSymbol. The possible values are:

- 0 = imsSolid
- 1 = imsDash
- 2 = imsDot
- 3 = imsDash\_dot
- 4 = imsDash\_dot\_dot

### **Syntax**

SimpleLineSymbol.Style

### **Arguments**

None

### **Returned Value**

imsLineStyle     Line style constant. The default value is imsSolid (0).

### **Example**

```
Dim mSimpleLineSymbol  
Set mSimpleLineSymbol = Server.CreateObject("aims.SimpleLineSymbol")  
mSimpleLineSymbol.Style = imsLineStyle.imsSolid
```

### **See Also**

imsLineStyle

## **SimpleLineSymbol.Transparency property**

### **Description**

The SimpleLineSymbol.Transparency property returns or sets the transparency coefficient for the SimpleLineSymbol. Lower numbers will display the symbol with greater transparency.

### **Syntax**

SimpleLineSymbol.Transparency

### **Arguments**

None

### **Returned Value**

Double Transparency coefficient. The default value is 1.0.

### **Example**

```
Dim mSimpleLineSymbol  
Set mSimpleLineSymbol = Server.CreateObject("aims.SimpleLineSymbol")  
mSimpleLineSymbol.Transparency = 1.0
```

## **SimpleLineSymbol.Width property**

### **Description**

The SimpleLineSymbol.Width property returns or sets the line width for a SimpleLineSymbol.

### **Syntax**

SimpleLineSymbol.Width

### **Arguments**

None

### **Returned Value**

Double      Line width.

### **Example**

```
Dim mSimpleLineSymbol  
Set mSimpleLineSymbol = Server.CreateObject("aims.SimpleLineSymbol")  
mSimpleLineSymbol.Width = 2
```

## **SimpleMarkerSymbol.Antialiasing property**

### **Description**

The SimpleMarkerSymbol.Antialiasing property returns or sets a Boolean value indicating whether or not antialiasing will be used while rendering the SimpleMarkerSymbol. Antialiasing allows lines to appear smooth by coloring adjacent pixels in such a way as to give the edge of a line a more gradual fade to the background.

### **Syntax**

SimpleMarkerSymbol.Antialising

### **Arguments**

None

### **Returned Value**

Boolean      Turns antialiasing on/off. The default value is False.

### **Example**

```
Dim mSimpleMarkerSymbol  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.SimpleMarkerSymbol")  
mSimpleMarkerSymbol.Antialising = True
```

## **SimpleMarkerSymbol.Clone method**

### **Description**

The SimpleMarkerSymbol.Clone method clones the SimpleMarkerSymbol.

### **Syntax**

SimpleMarkerSymbol.Clone ()

### **Arguments**

None

### **Returned Value**

SimpleMarkerSymbol      Cloned symbol.

### **Example**

```
dim mClone  
dim mSimpleMarkerSymbol  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.SimpleMarkerSymbol")  
Set mClone = mSimpleMarkerSymbol.Clone()
```

## **SimpleMarkerSymbol.Color property**

### **Description**

The SimpleMarkerSymbol.Color property returns or sets a symbol color constant for the SimpleMarkerSymbol.

### **Syntax**

SimpleMarkerSymbol.Color

### **Arguments**

None

### **Returned Value**

imsColor      Symbol color constant. The default value is imsBlack (0).

### **Example**

```
Dim mSimpleMarkerSymbol  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.SimpleMarkerSymbol")  
mSimpleMarkerSymbol.Color = imsColor.imsRed
```

### **See Also**

imsColor

## **SimpleMarkerSymbol.MarkerType property**

### **Description**

The SimpleMarkerSymbol.MarkerType property returns or sets a marker Symbol type constant for a SimpleMarkerSymbol. The possible values are:

- 0 = imsCircle
- 1 = imsTriangle
- 2 = imsSquare
- 3 = imsCross
- 4 = imsStar

### **Syntax**

SimpleMarkerSymbol.MarkerType

### **Arguments**

None

### **Returned Value**

imsMarkerType   Symbol type constant. The default value is imsCircle (0).

### **Example**

```
Dim mSimpleMarkerSymbol  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.SimpleMarkerSymbol")  
mSimpleMarkerSymbol.MarkerType = imsMarkerType.imsCross
```

### **See Also**

imsMarkerType

### **Note**

Also refer to the following samples:

Change\_marker\_symbol.asp

## **SimpleMarkerSymbol.Outline property**

### **Description**

The SimpleMarkerSymbol.Outline property returns or sets a color constant for the outline color of a SimpleMarkerSymbol.

### **Syntax**

SimpleMarkerSymbol.Outline

### **Arguments**

None

### **Returned Value**

imsColor      Outline color constant. The default value is imsBlack (0).

### **Example**

```
Dim mSimpleMarkerSymbol  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.SimpleMarkerSymbol")  
mSimpleMarkerSymbol.Outline = imsColor.imsRed
```

### **Note**

Also refer to the following sample:

[Change\\_marker\\_symbol.asp](#)

## **SimpleMarkerSymbol.Overlap property**

### **Description**

The SimpleMarkerSymbol.Overlap property returns or sets a Boolean value indicating whether or not labels will overlap a SimpleMarkerSymbol.

### **Syntax**

SimpleMarkerSymbol.Overlap

### **Arguments**

None

### **Returned Value**

Boolean      True indicates that labels will be allowed to overlap the SimpleMarkerSymbol. False indicates that labels will not overlap the symbol. The default value is True.

### **Example**

```
Dim mSimpleMarkerSymbol  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.SimpleMarkerSymbol")  
mSimpleMarkerSymbol.Overlap = False
```

## **SimpleMarkerSymbol.Shadow property**

### **Description**

The SimpleMarkerSymbol.Shadow property returns or sets a color constant for the shadow color of a SimpleMarkerSymbol.

### **Syntax**

SimpleMarkerSymbol.Shadow

### **Arguments**

None

### **Returned Value**

imsColor      Shadow color constant. The default value is imsBlack (0).

### **Example**

```
Dim mSimpleMarkerSymbol  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.SimpleMarkerSymbol")  
mSimpleMarkerSymbol.Shadow = imsColor.imsBlack
```

## **SimpleMarkerSymbol.Transparency property**

### **Description**

The SimpleMarkerSymbol.Transparency property returns or sets the transparency coefficient for the SimpleMarkerSymbol. Lower numbers will display the symbol with greater transparency.

### **Syntax**

SimpleMarkerSymbol.Transparency

### **Arguments**

None

### **Returned Value**

Double Transparency coefficient. The default value is 1.0.

### **Example**

```
Dim mSimpleMarkerSymbol  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.SimpleMarkerSymbol")  
mSimpleMarkerSymbol.Transparency = 1.0
```

## **SimpleMarkerSymbol.Width property**

### **Description**

The SimpleMarkerSymbol.Width property returns or sets the width of a SimpleMarkerSymbol.

### **Syntax**

SimpleMarkerSymbol.Width

### **Arguments**

None

### **Returned Value**

Double      Marker width.

### **Example**

```
Dim mSimpleMarkerSymbol  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.SimpleMarkerSymbol")  
mSimpleMarkerSymbol.Width = 2
```

## **SimplePolygonSymbol.Antialiasing property**

### **Description**

The SimplePolygonSymbol.Antialiasing property returns or sets a Boolean value indicating whether or not antialiasing will be used while rendering the SimplePolygonSymbol. Antialiasing allows lines to appear smooth by coloring adjacent pixels in such a way as to give the edge of a line a more gradual fade to the background.

### **Syntax**

SimplePolygonSymbol.Antialiasing

### **Arguments**

None

### **Returned Value**

Boolean      Turns antialiasing on/off. The default value is False.

### **Example**

```
Dim mSimplePolygonSymbol  
Set mSimplePolygonSymbol = Server.CreateObject("aims.SimplePolygonSymbol")  
mSimplePolygonSymbol.Antialiasing = True
```

## **SimplePolygonSymbol.Boundary property**

### **Description**

The SimplePolygonSymbol.Boundary property returns or sets a Boolean value that determines whether or not to draw a boundary symbol for the SimplePolygonSymbol.

### **Syntax**

SimplePolygonSymbol.Boundary

### **Arguments**

None

### **Returned Value**

Boolean      True indicates that a boundary will be drawn. The default value is true.

### **Example**

```
Dim mSimplePolygonSymbol  
Set mSimplePolygonSymbol = Server.CreateObject("aims.SimplePolygonSymbol")  
mSimplePolygonSymbol.Boundary = False
```

### **Note**

Also refer to the following sample:

[Change\\_polygon\\_symbol.asp](#)

## **SimplePolygonSymbol.BoundaryCapType** property

### **Description**

The SimplePolygonSymbol.BoundaryCapType property returns or sets a boundary line ending constant that determines how the boundary ends will be displayed. The possible values are:

0 = imsRound

1 = imsButt

2 = imsSquare

### **Syntax**

SimplePolygonSymbol.BoundaryCapType

### **Arguments**

None

### **Returned Value**

imsCapStyle      Boundary end type constant. The default value is imsButt (1).

### **Example**

```
Dim mSimplePolygonSymbol  
Set mSimplePolygonSymbol = Server.CreateObject("aims.SimplePolygonSymbol")  
mSimplePolygonSymbol.BoundaryCapType = imsCapStyle.imsRound
```

### **See Also**

imsCapStyle

## **SimplePolygonSymbol.BoundaryColor property**

### **Description**

The SimplePolygonSymbol.BoundaryColor property returns or sets a color constant for the boundary color of a SimplePolygonSymbol.

### **Syntax**

SimplePolygonSymbol.BoundaryColor

### **Arguments**

None

### **Returned Value**

imsColor      Boundary color constant. The default value is imsBlack (0).

### **Example**

```
Dim mSimplePolygonSymbol  
Set mSimplePolygonSymbol = Server.CreateObject("aims.SimplePolygonSymbol")  
mSimplePolygonSymbol.BoundaryColor = imsColor.imsBlack
```

### **Note**

Also refer to the following sample:

[Change\\_polygon\\_symbol.asp](#)

## **SimplePolygonSymbol.BoundaryStyle property**

### **Description**

The SimplePolygonSymbol.BoundaryStyle property returns or sets a constant that determines the boundary style for a SimplePolygonSymbol boundary. The possible values are:

- 0 = imsSolid
- 1 = imsDash
- 2 = imsDot
- 3 = imsDash\_dot
- 4 = imsDash\_dot\_dot

### **Syntax**

SimplePolygonSymbol.BoundaryStyle

### **Arguments**

None

### **Returned Value**

imsLineStyle      Boundary style constant. The default value is imsSolid (0).

### **Example**

```
Dim mSimplePolygonSymbol  
Set mSimplePolygonSymbol = Server.CreateObject("aims.SimplePolygonSymbol")  
mSimplePolygonSymbol.BoundaryStyle = imsLineStyle.imsSolid
```

### **See Also**

imsLineStyle

### **Note**

Also refer to samples:

Change\_Polygon\_Symbol.asp

## **SimplePolygonSymbol.BoundaryTransparency property**

### **Description**

The SimplePolygonSymbol.BoundaryTransparency property returns or sets the boundary transparency coefficient for the SimplePolygonSymbol. Lower numbers will display the symbol with greater transparency.

### **Syntax**

SimplePolygonSymbol.BoundaryTransparency

### **Arguments**

None

### **Returned Value**

Double      Boundary transparency coefficient. The default value is 1.0.

### **Example**

```
Dim mSimplePolygonSymbol  
Set mSimplePolygonSymbol = Server.CreateObject("aims.SimplePolygonSymbol")  
mSimplePolygonSymbol.BoundaryTransparency = 1.0
```

## **SimplePolygonSymbol.BoundaryWidth property**

### **Description**

The SimplePolygonSymbol.BoundaryWidth property returns or sets the boundary width of the SimplePolygonSymbol.

### **Syntax**

SimplePolygonSymbol.BoundaryWidth

### **Arguments**

None

### **Returned Value**

Double      Boundary width. The default value is 0.

### **Example**

```
Dim mSimplePolygonSymbol  
Set mSimplePolygonSymbol = Server.CreateObject("aims.SimplePolygonSymbol")  
mSimplePolygonSymbol.BoundaryWidth = 1
```

### **Note**

Also refer to samples:  
[Change\\_Polygon\\_Symbol.asp](#)

## **SimplePolygonSymbol.Clone method**

### **Description**

The SimplePolygonSymbol.Clone method clones the SimplePolygonSymbol.

### **Syntax**

SimplePolygonSymbol.Clone ()

### **Arguments**

None

### **Returned Value**

SimplePolygonSymbol    Cloned symbol.

### **Example**

```
dim mClone  
dim mSimplePolygonSymbol  
Set mSimplePolygonSymbol = Server.CreateObject("aims.SimplePolygonSymbol")  
Set mClone = mSimplePolygonSymbol.Clone()
```

## **SimplePolygonSymbol.FillColor property**

### **Description**

The SimplePolygonSymbol.FillColor property returns or sets a color constant for the fill color of a SimplePolygonSymbol.

### **Syntax**

SimplePolygonSymbol.FillColor

### **Arguments**

None

### **Returned Value**

imsColor      The fill color constant. The default value is imsBlack (0).

### **Example**

```
Dim mSimplePolygonSymbol  
Set mSimplePolygonSymbol = Server.CreateObject("aims.SimplePolygonSymbol")  
mSimplePolygonSymbol.FillColor = imsColor.imsRed
```

### **Note**

Also refer to samples:

[Change\\_Polygon\\_Symbol.asp](#)

## **SimplePolygonSymbol.FillInterval property**

### **Description**

The SimplePolygonSymbol.FillInterval property returns or sets the fill interval for hatch fills.

### **Syntax**

SimplePolygonSymbol.FillInterval

### **Arguments**

None

### **Returned Value**

Double      Fill interval for hatch fills. The default value is 6.

### **Example**

```
Dim mSimplePolygonSymbol  
Set mSimplePolygonSymbol = Server.CreateObject("aims.SimplePolygonSymbol")  
mSimplePolygonSymbol.FillInterval = 10
```

### **Note**

Also refer to samples:

Change\_Polygon\_Symbol.asp

## **SimplePolygonSymbol.FillStyle property**

### **Description**

The SimplePolygonSymbol.FillStyle property returns or sets a constant indicating the fill style for the SimplePolygonSymbol. The possible values are:

- 0 = imsSolid
- 1 = imsBDiagonal
- 2 = imsFDiagonal
- 3 = imsCross
- 4 = imsDiagcross
- 5 = imsHorizontal
- 6 = imsVertical
- 7 = imsLightGray
- 8 = imsGray
- 9 = imsDarkGray

### **Syntax**

SimplePolygonSymbol.FillStyle

### **Arguments**

None

### **Returned Value**

imsFillStyle      Symbol fill style constant. The default value is imsSolid (0).

### **Example**

```
Dim mSimplePolygonSymbol  
Set mSimplePolygonSymbol = Server.CreateObject("aims.SimplePolygonSymbol")  
mSimplePolygonSymbol.FillStyle=imsFillStyle.imsSolid
```

### **See Also**

imsFillStyle

### **Note**

Also refer to samples:

[Change\\_Polygon\\_Symbol.asp](#)

## **SimplePolygonSymbol.JoinStyle** property

### **Description**

The SimplePolygonSymbol.JoinStyle property returns or sets a join style constant that determines how intersecting (or joined) boundary lines are displayed with SimplePolygonSymbols. The possible values are:

- 0 = imsRound
- 1 = imsMiter
- 2 = imsBevel

### **Syntax**

SimplePolygonSymbol.JoinStyle

### **Arguments**

None

### **Returned Value**

imsJoinStyle     Line join style constant. The default value is imsRound (0).

### **Example**

```
Dim mSimplePolygonSymbol  
Set mSimplePolygonSymbol = Server.CreateObject("aims.SimplePolygonSymbol")  
mSimplePolygonSymbol.JoinStyle = imsJoinStyle.imsRound
```

### **See Also**

imsJoinStyle

## **SimpleRenderer.Clone method**

### **Description**

The SimpleRenderer.Clone method clones the SimpleRenderer.

### **Syntax**

SimpleRenderer.Clone()

### **Arguments**

None

### **Returned Value**

SimpleRenderer      Cloned renderer.

### **Example**

```
Dim mClone  
Dim mSimpleRenderer  
Set mSimpleRenderer = Server.CreateObject("aims.SimpleRendererr")  
Set mClone = mSimpleRenderer.Clone()
```

## **SimpleRenderer.Symbol** property

### **Description**

The SimpleRenderer.Symbol property returns or sets a symbol object for a renderer. It can be a TrueTypeMarkerSymbol, SimpleMarkerSymbol, RasterMarkerSymbol, HashLineSymbol, SimpleLineSymbol, RasterFillSymbol, SimplePolygonSymbol, or a GradientFillSymbol.

### **Syntax**

SimpleRenderer.Symbol

### **Arguments**

None

### **Returned Value**

Object              Symbol for a renderer.

### **Example**

```
Dim mSimpleRenderer  
Dim mSimpleMarkerSymbol  
Set mSimpleRenderer = Server.CreateObject("aims.SimpleRenderer")  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.SimpleMarkerSymbol")  
mSimpleRenderer.Symbol = mSimpleMarkerSymbol
```

### **See Also**

TrueTypeMarkerSymbol  
SimpleMarkerSymbol  
RasterMarkerSymbol  
HashLineSymbol  
SimpleLineSymbol  
RasterFillSymbol  
SimplePolygonSymbol  
GradientFillSymbol

### **Note**

Also refer to the following samples:

Change\_marker\_symbol.asp  
Change\_polygon\_symbol.asp  
Buffer.asp

## **TableDesc.Count property**

### **Description**

The TableDesc.Count property returns the number of described fields.

### **Syntax**

TableDesc.Count

### **Arguments**

None

### **Returned Value**

Long            None            Number of the fields.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mTableDesc  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSServer.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
Set mTableDesc = mMap.Layers.Item(2).Recordset.TableDesc  
mCount = mTableDesc.Count  
Response.write mCount
```

### **Note**

Also refer to samples:

Sample10.asp

## TableDesc.FieldLength method

### Description

The TableDesc.FieldLength method returns the length of the field by index.

### Syntax

TableDesc.FieldLength(index)

### Arguments

index	Long	None	Index to reference to the field.
-------	------	------	----------------------------------

### Returned Value

Long	None	Length of the field.
------	------	----------------------

### Example

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mTableDesc
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
mMap.Init mArcIMSConnector, "IMSMAPService"
Set mTableDesc = mMap.Layers.Item(2).Recordset.TableDesc
```

```
for I = 1 to mTableDesc.Count
    mFieldLength = mTableDesc.FieldLength(i)
next
```

### See Also

[TableDesc.FieldName](#)

[TableDescFieldType](#)

[TableDesc.FieldPrecision](#)

## **TableDesc.FieldName method**

### **Description**

The TableDesc.FieldName method returns the name of the field by index.

### **Syntax**

TableDesc.FieldName(index)

### **Arguments**

index	Long	None	Index to reference to the field.
-------	------	------	----------------------------------

### **Returned Value**

String	None	Name of the field.
--------	------	--------------------

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mTableDesc
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mTableDesc = mMap.Layers.Item(2).Recordset.TableDesc
```

```
For I = 1 to mTableDesc.Count
```

```
    mTableDesc.FieldName(i)
```

```
Next
```

### **Note**

Also refer to samples:

Sample10.asp

## TableDesc.FieldPrecision method

### Description

The TableDesc.FieldPrecision method returns the precision of the field by index.

### Syntax

TableDesc.FieldPrecision(index)

### Arguments

index	Long	None	Index to reference to the field.
-------	------	------	----------------------------------

### Returned Value

Long	None	Precision of the field.
------	------	-------------------------

### See Also

TableDesc.FieldLength

TableDesc.FieldName

TableDesc.FieldType

### Example

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mTableDesc
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
mArcIMSConnector.ServerName = "IMSServer"
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mTableDesc = mMap.Layers.Item(2).Recordset.TableDesc
```

```
For I = 1 to mTableDesc.Count
```

```
    mFieldPrecision = mTableDesc.FieldPrecision(i)
```

```
Next
```

## **TableDesc.FieldType method**

### **Description**

The TableDesc.FieldType method returns the type of the field by index.

### **Syntax**

TableDesc.FieldType(index)

### **Arguments**

index	Long	None	Index to reference to the field.
-------	------	------	----------------------------------

### **Returned Value**

Long	None	Type of the field. Use imsFieldType constants.
------	------	--

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mTableDesc
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
```

```
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
```

```
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set mTableDesc = mMap.Layers.Item(2).Recordset.TableDesc
```

```
For I = 1 to mTableDesc.Count
```

```
    mTableDesc.FieldType(i)
```

```
Next
```

### **See Also**

[TableDesc.FieldLength](#)

[TableDesc.FieldName](#)

[TableDesc.FieldPrecision](#)

[imsFieldType](#)

## **TextMarkerSymbol.Angle property**

### **Description**

The TextMarkerSymbol.Angle property returns or sets the angle of rotation for a TextMarkerSymbol in degrees going counterclockwise; 0 degrees is horizontal.

### **Syntax**

TextMarkerSymbol.Angle

### **Arguments**

None

### **Returned Value**

Double      Angle of rotation. The default value is 0.

### **Example**

```
Dim mTextMarkerSymbol  
Set mTextMarkerSymbol = Server.CreateObject("aims.TextMarkerSymbol")  
mTextMarkerSymbol.Angle = 30
```

## **TextMarkerSymbol.Antialiasing property**

### **Description**

The TextMarkerSymbol.Antialiasing property returns or sets a Boolean value indicating whether or not antialiasing will be used while rendering the TextMarkerSymbol. Antialiasing allows lines to appear smooth by coloring adjacent pixels in such a way as to give the edge of a line a more gradual fade to the background.

### **Syntax**

TextMarkerSymbol.Antialiasing

### **Arguments**

None

### **Returned Value**

Boolean      Turns antialiasing on/off. The default value is False.

### **Example**

```
Dim mTextMarkerSymbol  
Set mTextMarkerSymbol = Server.CreateObject("aims.TextMarkerSymbol")  
mTextMarkerSymbol.Antialiasing = True
```

## **TextMarkerSymbol.Blockout property**

### **Description**

The TextMarkerSymbol.Blockout property returns or sets a constant for the background color of a TextMarkerSymbol.

### **Syntax**

TextMarkerSymbol.Blockout

### **Arguments**

None

### **Returned Value**

imsColor              Background color constant. The default value is imsBlack (0).

### **Example**

```
Dim mTextMarkerSymbol  
Set mTextMarkerSymbol = Server.CreateObject("aims.TextMarkerSymbol")  
mTextMarkerSymbol.Blockout = imsColor.imsRed
```

### **See Also**

imsColor

## **TextMarkerSymbol.Clone method**

### **Description**

The TextMarkerSymbol.Clone method clones the TextMarkerSymbol.

### **Syntax**

TextMarkerSymbol.Clone ()

### **Arguments**

None

### **Returned Value**

TextMarkerSymbol      Cloned symbol.

### **Example**

```
dim mClone  
dim mTextMarkerSymbol  
Set mTextMarkerSymbol = Server.CreateObject("aims.TextMarkerSymbol")  
Set mClone = mTextMarkerSymbol.Clone()
```

## **TextMarkerSymbol.Font property**

### **Description**

The TextMarkerSymbol.Font property returns or sets the name of the font used with the TextMarkerSymbol.

### **Syntax**

TextMarkerSymbol.Font

### **Arguments**

None

### **Returned Value**

String            Font name. The default value is “default”

### **Example**

```
Dim mTextMarkerSymbol  
Set mTextMarkerSymbol = Server.CreateObject("aims.TextMarkerSymbol")  
mTextMarkerSymbol.Font = "Arial"
```

## **TextMarkerSymbol.FontColor property**

### **Description**

The TextMarkerSymbol.FontColor property returns or sets a font color constant for the font used with a TextMarkerSymbol.

### **Syntax**

TextMarkerSymbol.Color

### **Arguments**

None

### **Returned Value**

imsColor      Font color. The default value is imsBlack (0).

### **Example**

```
Dim mTextMarkerSymbol  
Set mTextMarkerSymbol = Server.CreateObject("aims.TextMarkerSymbol")  
mTextMarkerSymbol.Color = imsColor.imsRed
```

## **TextMarkerSymbol.FontSize property**

### **Description**

The TextMarkerSymbol.FontSize property returns or sets the font size for the font used with a TextMarkerSymbol.

### **Syntax**

TextMarkerSymbol.FontSize

### **Arguments**

None

### **Returned Value**

Long            Font size. The default value is 12.

### **Example**

```
Dim mTextMarkerSymbol  
Set mTextMarkerSymbol = Server.CreateObject("aims.TextMarkerSymbol")  
mTextMarkerSymbol.FontSize = 10
```

## **TextMarkerSymbol.FontStyle property**

### **Description**

The TextMarkerSymbol.FontStyle property returns or sets a font style constant for the font used with a TextMarkerSymbol. The possible values are:

0 = imsRegular

1 = imsBold

2 = imsItalic

3 = imsUnderline

4 = imsOutline

### **Syntax**

TextMarkerSymbol.FontStyle

### **Arguments**

None

### **Returned Value**

imsFontStyle      Font style. The default value is imsRegular (0).

### **Example**

```
Dim mTextMarkerSymbol  
Set mTextMarkerSymbol = Server.CreateObject("aims.TextMarkerSymbol")  
mTextMarkerSymbol.FontStyle = imsFontStyle.imsBold
```

## **TextMarkerSymbol.Glowing property**

### **Description**

The TextMarkerSymbol.Glowing property returns or sets the glowing color constant for a TextMarkerSymbol. Glowing is the halo effect around text labels.

### **Syntax**

TextMarkerSymbol.Glowing

### **Arguments**

None

### **Returned Value**

imsColor      Glowing color. The default value is imsBlack (0).

### **Example**

```
Dim mTextMarkerSymbol  
Set mTextMarkerSymbol = Server.CreateObject("aims.TextMarkerSymbol")  
mTextMarkerSymbol.Glowing = imsColor.imsRed
```

## **TextMarkerSymbol.HAlignment property**

### **Description**

The TextMarkerSymbol.HAlignment property returns or sets a constant indicating the horizontal alignment of label compared to the label point. The possible values are:

0 = imsLeft

1 = imsCenter

2 = imsRight

### **Syntax**

TextMarkerSymbol.HAlignment

### **Arguments**

None

### **Returned Value**

imsHAlignment Horizontal alignment constant. The default value is imsRight (2).

### **Example**

```
Dim mTextMarkerSymbol  
Set mTextMarkerSymbol = Server.CreateObject("aims.TextMarkerSymbol")  
mTextMarkerSymbol.HAlignment= imsHAlignment.imsLeft
```

### **See Also**

imsHAlignment

## **TextMarkerSymbol.Interval property**

### **Description**

The TextMarkerSymbol.Interval property returns or sets the distance (pixels) between a feature and the printed label.

### **Syntax**

TextMarkerSymbol.Interval

### **Arguments**

None

### **Returned Value**

Double      Distance between point and printed label. The default value is 0.

### **Example**

```
Dim mTextMarkerSymbol  
Set mTextMarkerSymbol = Server.CreateObject("aims.TextMarkerSymbol")  
mTextMarkerSymbol.Interval = 10
```

## **TextMarkerSymbol.Outline property**

### **Description**

The TextMarkerSymbol.Outline property returns or sets a color constant for the outline color of a TextMarkerSymbol.

### **Syntax**

TextMarkerSymbol.Outline

### **Arguments**

None

### **Returned Value**

imsColor      Outline color constant. The default value is imsBlack (0).

### **Example**

```
Dim mTextMarkerSymbol  
Set mTextMarkerSymbol = Server.CreateObject("aims.TextMarkerSymbol")  
mTextMarkerSymbol.Outline = imsColor imsRed
```

## **TextMarkerSymbol.PrintMode property**

### **Description**

The TextMarkerSymbol.PrintMode property returns or sets a PrintMode constant that determines how the text for a TextMarkerSymbol will be rendered. The possible values are:

- 0 = imsNone
- 1 = imsTitleCaps
- 2 = imsAllUpper
- 3 = imsAllLower

### **Syntax**

`TextMarkerSymbol.PrintMode`

### **Arguments**

None

### **Returned Value**

`imsPrintMode` Printing mode constant. The default value is imsNone (0).

### **Example**

```
Dim mTextMarkerSymbol  
Set mTextMarkerSymbol = Server.CreateObject("aims.TextMarkerSymbol")  
mTextMarkerSymbol.PrintMode = imsPrintMode.imsNone
```

### **See Also**

`imsPrintMode`

## **TextMarkerSymbol.Shadow property**

### **Description**

The TextMarkerSymbol.Shadow property returns or sets a color constant for the shadow color of a TextMarkerSymbol.

### **Syntax**

TextMarkerSymbol.Shadow

### **Arguments**

None

### **Returned Value**

imsColor      Shadow color constant. The default value is imsBlack (0).

### **Example**

```
Dim mTextMarkerSymbol  
Set mTextMarkerSymbol = Server.CreateObject("aims.TextMarkerSymbol")  
mTextMarkerSymbol.Shadow = imsColor.imsBlack
```

## **TextMarkerSymbol.Transparency** property

### **Description**

The TextMarkerSymbol.Transparency property returns or sets the transparency coefficient for the TextMarkerSymbol. Lower numbers will display the symbol with greater transparency.

### **Syntax**

TextMarkerSymbol.Transparency

### **Arguments**

None

### **Returned Value**

Double Transparency coefficient. The default value is 1.0.

### **Example**

```
Dim mTextMarkerSymbol  
Set mTextMarkerSymbol = Server.CreateObject("aims.TextMarkerSymbol")  
mTextMarkerSymbol.Transparency = 1.0
```

## **TextMarkerSymbol.VAlignment property**

### **Description**

The TextMarkerSymbol.VAlignment property returns or sets a constant indicating the vertical alignment of a TextMarkerSymbol compared to a label point. The possible values are:

- 0 = imsTop
- 1 = imsCenter
- 2 = imsBottom

### **Syntax**

TextMarkerSymbol.VAlignment

### **Arguments**

None

### **Returned Value**

imsVAlignment Vertical alignment constant. The default value is imsTop (0).

### **Example**

```
Dim mTextMarkerSymbol  
Set mTextMarkerSymbol = Server.CreateObject( "aims.TextMarkerSymbol" )  
mTextMarkerSymbol.VAlignment= imsVAlignment.imsTop
```

### **See Also**

imsVAlignment

## **TextObject.Id method**

### **Description**

The TextObject.Id method contains the id for the text object.

### **Syntax**

TextObject.Id

### **Arguments**

None

### **Returned Value**

String            “Text”            Text Object’s id.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mTableDesc  
Dim al  
Dim text  
Dim mId  
  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.Visible = true  
  
Set text = Server.CreateObject("aims.TextObject")  
mId = text.Id  
Response.write mId
```

## **TextObject.Label property**

### **Description**

The TextObject.Label property contains the label for the text object.

### **Syntax**

TextObject.Label

### **Arguments**

None

### **Returned Value**

String               “”               Text to add to label.

### **Example**

Dim mArcIMSConnector

Dim mMap

Dim mTableDesc

Dim al

Dim text

Dim mId

Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")

mArcIMSConnector.ServerName = "IMSServer"

mArcIMSConnector.ServerPort = 5300

Set mMap = Server.CreateObject("aims.Map")

mMap.InitMap mArcIMSConnector, "IMSMAPService"

Set al = Server.CreateObject("aims.AcetateLayer")

al.Visible = true

Set text = Server.CreateObject("aims.TextObject")

text.Label = "ActiveXConnector"

## **TextObject.Name property**

### **Description**

The TextObject.Name property returns or sets the text object's name property.

### **Syntax**

TextObject.Name

### **Arguments**

None

### **Returned Value**

String            “Text”            TextObject's name.

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mTableDesc  
Dim al  
Dim text  
Dim mId  
  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.Visible = true  
  
Set text = Server.CreateObject("aims.TextObject")  
text.Name = "ActiveXConnector"
```

## **TextObject.TextMarkerSymbol property**

### **Description**

The TextObject.TextMarkerSymbol property specifies how to depict text.

### **Syntax**

TextObject.TextMarkerSymbol

### **Arguments**

None

### **Returned Value**

TextMarkerSymbol      None      TextMarkerSymbol object.

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mTableDesc
```

```
Dim al
```

```
Dim text
```

```
Dim mId
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
```

```
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
```

```
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set al = Server.CreateObject("aims.AcetateLayer")
```

```
al.Visible = true
```

```
Set text = Server.CreateObject("aims.TextObject")
```

```
text.TextMarkerSymbol.Color = 255
```

## **TextObject.X property**

### **Description**

The TextObject.X property contains the X coordinate of the text.

### **Syntax**

TextObject.X

### **Arguments**

None

### **Returned Value**

Double            0            X coordinate

### **Example**

```
Dim mArcIMSConnector  
Dim mMap  
Dim mTableDesc  
Dim al  
Dim text  
Dim mId  
  
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")  
mArcIMSConnector.ServerName = "IMSServer"  
mArcIMSConnector.ServerPort = 5300  
Set mMap = Server.CreateObject("aims.Map")  
mMap.InitMap mArcIMSConnector, "IMSMAPService"  
  
Set al = Server.CreateObject("aims.AcetateLayer")  
al.Visible = true  
  
Set text = Server.CreateObject("aims.TextObject")  
text.X = 56
```

## **TextObject.Y property**

### **Description**

The TextObject.Y property contains the Y coordinate of the text.

### **Syntax**

TextObject.Y

### **Arguments**

None

### **Returned Value**

Double        0        Y coordinate.

### **Example**

```
Dim mArcIMSConnector
```

```
Dim mMap
```

```
Dim mTableDesc
```

```
Dim al
```

```
Dim text
```

```
Dim mId
```

```
Set mArcIMSConnector = Server.CreateObject("aims.ArcIMSConnector")
```

```
mArcIMSConnector.ServerName = "IMSServer"
```

```
mArcIMSConnector.ServerPort = 5300
```

```
Set mMap = Server.CreateObject("aims.Map")
```

```
mMap.InitMap mArcIMSConnector, "IMSMAPService"
```

```
Set al = Server.CreateObject("aims.AcetateLayer")
```

```
al.Visible = true
```

```
Set text = Server.CreateObject("aims.TextObject")
```

```
text.Y = 46
```

## **TextSymbol.Antialiasing property**

### **Description**

The TextSymbol.Antialiasing property returns or sets a Boolean value indicating whether or not antialiasing will be used while rendering the TextSymbol. Antialiasing allows lines to appear smooth by coloring adjacent pixels in such a way as to give the edge of a line a more gradual fade to the background.

### **Syntax**

TextSymbol.Antialiasing

### **Arguments**

None

### **Returned Value**

Boolean      Turns antialiasing on/off. The default value is False.

### **Example**

```
Dim mTextSymbol  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mTextSymbol.Antialiasing = True
```

## **TextSymbol.Blockout property**

### **Description**

The TextSymbol.Blockout property returns or sets a constant for the background color of a TextSymbol.

### **Syntax**

TextSymbol.Blockout

### **Arguments**

None

### **Returned Value**

imsColor      Background color constant. The default value is imsBlack (0).

### **Example**

```
Dim mTextSymbol  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mTextSymbol.Blockout = imsColor imsRed
```

### **See Also**

imsColor

## **TextSymbol.Clone method**

### **Description**

The TextSymbol.Clone method clones the TextSymbol.

### **Syntax**

TextSymbol.Clone ()

### **Arguments**

None

### **Returned Value**

TextSymbol      Cloned symbol.

### **Example**

```
dim mClone  
dim mTextSymbol  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
Set mClone = mTextSymbol.Clone()
```

## **TextSymbol.Font property**

### **Description**

The TextSymbol.Font property returns or sets the name of the font used with the TextSymbol.

### **Syntax**

TextSymbol.Font

### **Arguments**

None

### **Returned Value**

String            Font name. The default value is “default”

### **Example**

```
Dim mTextSymbol  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mTextSymbol.Font = "Arial"
```

## **TextSymbol.FontColor property**

### **Description**

The TextSymbol.FontColor property returns or sets a font color constant for the font used with a TextSymbol.

### **Syntax**

TextSymbol.Color

### **Arguments**

None

### **Returned Value**

imsColor      Font color. The default value is imsBlack (0).

### **Example**

```
Dim mTextSymbol  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mTextSymbol.Color = imsColor.imsRed
```

## **TextSymbol.FontSize property**

### **Description**

The TextSymbol.FontSize property returns or sets the font size for the font used with a TextSymbol.

### **Syntax**

TextSymbol.FontSize

### **Arguments**

None

### **Returned Value**

Long            Font size. The default value is 12.

### **Example**

```
Dim mTextSymbol  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mTextSymbol.FontSize = 10
```

## **TextSymbol.FontStyle property**

### **Description**

The TextSymbol.FontStyle property returns or sets a font style constant for the font used with a TextSymbol. The possible values are:

- 0 = imsRegular
- 1 = imsBold
- 2 = imsItalic
- 3 = imsUnderline
- 4 = imsOutline

### **Syntax**

TextSymbol.FontStyle

### **Arguments**

None

### **Returned Value**

imsFontStyle      Font style. The default value is imsRegular (0).

### **Example**

```
Dim mTextSymbol  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mTextSymbol.FontStyle = imsFontStyle.imsBold
```

## **TextSymbol.Glowing property**

### **Description**

The TextSymbol.Glowing property returns or sets the glowing color constant for a TextSymbol. Glowing is the halo effect around text labels.

### **Syntax**

TextSymbol.Glowing

### **Arguments**

None

### **Returned Value**

imsColor      Glowing color. The default value is imsBlack (0).

### **Example**

```
Dim mTextSymbol  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mTextSymbol.Glowing = imsColor.imsRed
```

## **TextSymbol.Interval property**

### **Description**

The TextSymbol.Interval property returns or sets the distance (pixels) between a feature and the printed label.

### **Syntax**

TextSymbol.Interval

### **Arguments**

None

### **Returned Value**

Double      Distance between point and printed label. The default value is 0.

### **Example**

```
Dim mTextSymbol  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mTextSymbol.Interval = 10
```

## **TextSymbol.Outline property**

### **Description**

The TextSymbol.Outline property returns or sets a color constant for the outline color of a TextSymbol.

### **Syntax**

TextSymbol.Outline

### **Arguments**

None

### **Returned Value**

imsColor      Outline color constant. The default value is imsBlack (0).

### **Example**

```
Dim mTextSymbol  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mTextSymbol.Outline = imsColor.imsRed
```

## **TextSymbol.PrintMode property**

### **Description**

The TextSymbol.PrintMode property returns or sets a PrintMode constant that determines how the text for a TextSymbol will be rendered. The possible values are:

- 0 = imsNone
- 1 = imsTitleCaps
- 2 = imsAllUpper
- 3 = imsAllLower

### **Syntax**

`TextSymbol.PrintMode`

### **Arguments**

None

### **Returned Value**

`imsPrintMode` Printing mode constant. The default value is imsNone (0).

### **Example**

```
Dim mTextSymbol  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mTextSymbol.PrintMode = imsPrintMode.imsNone
```

### **See Also**

`imsPrintMode`

## **TextSymbol.Shadow property**

### **Description**

The TextSymbol.Shadow property returns or sets a color constant for the shadow color of a TextSymbol.

### **Syntax**

TextSymbol.Shadow

### **Arguments**

None

### **Returned Value**

imsColor      Shadow color constant. The default value is imsBlack (0).

### **Example**

```
Dim mTextSymbol  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mTextSymbol.Shadow = imsColor.imsBlack
```

## **TextSymbol.Transparency property**

### **Description**

The TextSymbol.Transparency property returns or sets the transparency coefficient for the TextSymbol. Lower numbers will display the symbol with greater transparency.

### **Syntax**

TextSymbol.Transparency

### **Arguments**

None

### **Returned Value**

Double Transparency coefficient. The default value is 1.0.

### **Example**

```
Dim mTextSymbol  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mTextSymbol.Transparency = 1.0
```

## **TrueTypeMarkerSymbol.Angle property**

### **Description**

The TrueTypeMarkerSymbol.Angle property returns or sets the angle of rotation for a TrueTypeMarkerSymbol in degrees going counterclockwise; 0 degrees is horizontal.

### **Syntax**

TrueTypeMarkerSymbol.Angle

### **Arguments**

None

### **Returned Value**

Double            Angle of rotation. The default value is 0.

### **Example**

```
Dim mTrueTypeMarkerSymbol  
Set mTrueTypeMarkerSymbol = Server.CreateObject("aims.TrueTypeMarkerSymbol")  
mTrueTypeMarkerSymbol.Angle = 30
```

## **TrueTypeMarkerSymbol.Antialiasing property**

### **Description**

The TrueTypeMarkerSymbol.Antialiasing property returns or sets a Boolean value indicating whether or not antialiasing will be used while rendering the TrueTypeMarkerSymbol. Antialiasing allows lines to appear smooth by coloring adjacent pixels in such a way as to give the edge of a line a more gradual fade to the background.

### **Syntax**

TrueTypeMarkerSymbol.Antialiasing

### **Arguments**

None

### **Returned Value**

Boolean      Turns antialiasing on/off. The default value is False.

### **Example**

```
Dim mTrueTypeMarkerSymbol  
Set mTrueTypeMarkerSymbol = Server.CreateObject("aims.TrueTypeMarkerSymbol")  
mTrueTypeMarkerSymbol.Antialiasing = True
```

## **TrueTypeMarkerSymbol.Character** property

### **Description**

The TrueTypeMarkerSymbol.Character property returns or sets a text character index in a font containing the symbol to use in a decimal notation. Available values are 32–255.

### **Syntax**

TrueTypeMarkerSymbol.Character

### **Arguments**

None

### **Returned Value**

Long            ASCII code of the text character in font containing symbol to use in a decimal notation.

### **Example**

```
Dim mTrueTypeMarkerSymbol  
Set mTrueTypeMarkerSymbol = Server.CreateObject("aims.TrueTypeMarkerSymbol")  
mTrueTypeMarkerSymbol.Character = 35
```

## **TrueTypeMarkerSymbol.Clone method**

### **Description**

The TrueTypeMarkerSymbol.Clone method clones the TrueTypeMarkerSymbol.

### **Syntax**

TrueTypeMarkerSymbol.Clone ()

### **Arguments**

None

### **Returned Value**

TrueTypeMarkerSymbol Cloned symbol.

### **Example**

```
dim mClone  
dim mTrueTypeMarkerSymbol  
Set mTrueTypeMarkerSymbol = Server.CreateObject("aims.TrueTypeMarkerSymbol")  
Set mClone = mTrueTypeMarkerSymbol.Clone()
```

## **TrueTypeMarkerSymbol.Font** property

### **Description**

The TrueTypeMarkerSymbol.Font property returns or sets the name of the font used with the TrueTypeMarkerSymbol.

### **Syntax**

TrueTypeMarkerSymbol.Font

### **Arguments**

None

### **Returned Value**

String            Font name. The default value is “default”.

### **Example**

```
Dim mTrueTypeMarkerSymbol  
Set mTrueTypeMarkerSymbol = Server.CreateObject("aims.TrueTypeMarkerSymbol")  
mTrueTypeMarkerSymbol.Font = "Arial"
```

## **TrueTypeMarkerSymbol.FontColor property**

### **Description**

The TrueTypeMarkerSymbol.FontColor property returns or sets a font color constant for the font used with a TrueTypeMarkerSymbol.

### **Syntax**

TrueTypeMarkerSymbol.Color

### **Arguments**

None

### **Returned Value**

imsColor      Font color. The default value is imsBlack (0).

### **Example**

```
Dim mTrueTypeMarkerSymbol  
Set mTrueTypeMarkerSymbol = Server.CreateObject("aims.TrueTypeMarkerSymbol")  
mTrueTypeMarkerSymbol.Color = imsColor.imsRed
```

## **TrueTypeMarkerSymbol.FontSize property**

### **Description**

The TrueTypeMarkerSymbol.FontSize property returns or sets the font size for the font used with a TrueTypeMarkerSymbol.

### **Syntax**

TrueTypeMarkerSymbol.FontSize

### **Arguments**

None

### **Returned Value**

Long            Font size. The default value is 12.

### **Example**

```
Dim mTrueTypeMarkerSymbol  
Set mTrueTypeMarkerSymbol = Server.CreateObject("aims.TrueTypeMarkerSymbol")  
mTrueTypeMarkerSymbol.FontSize = 10
```

## **TrueTypeMarkerSymbol.FontStyle property**

### **Description**

The TrueTypeMarkerSymbol.FontStyle property returns or sets a font style constant for the font used with a TrueTypeMarkerSymbol. The possible values are:

- 0 = imsRegular
- 1 = imsBold
- 2 = imsItalic
- 3 = imsUnderline
- 4 = imsOutline

### **Syntax**

TrueTypeMarkerSymbol.FontStyle

### **Arguments**

None

### **Returned Value**

imsFontStyle      Font style. The default value is imsRegular (0).

### **Example**

```
Dim mTrueTypeMarkerSymbol  
Set mTrueTypeMarkerSymbol = Server.CreateObject("aims.TrueTypeMarkerSymbol")  
mTrueTypeMarkerSymbol.FontStyle = imsFontStyle.imsBold
```

## **TrueTypeMarkerSymbol.Glowing property**

### **Description**

The TrueTypeMarkerSymbol.Glowing property returns or sets the glowing color constant for a TrueTypeMarkerSymbol. Glowing is the halo effect around the symbol.

### **Syntax**

TrueTypeMarkerSymbol.Glowing

### **Arguments**

None

### **Returned Value**

imsColor      Glowing color. The default value is imsBlack (0).

### **Example**

```
Dim mTrueTypeMarkerSymbol  
Set mTrueTypeMarkerSymbol = Server.CreateObject("aims.TrueTypeMarkerSymbol")  
mTrueTypeMarkerSymbol.Glowing = imsColor imsRed
```

## **TrueTypeMarkerSymbol.Outline property**

### **Description**

The TrueTypeMarkerSymbol.Outline property returns or sets a color constant for the outline color of a TrueTypeMarkerSymbol.

### **Syntax**

TrueTypeMarkerSymbol.Outline

### **Arguments**

None

### **Returned Value**

imsColor      Outline color constant. The default value is imsBlack (0).

### **Example**

```
Dim mTrueTypeMarkerSymbol  
Set mTrueTypeMarkerSymbol = Server.CreateObject("aims.TrueTypeMarkerSymbol")  
mTrueTypeMarkerSymbol.Outline = imsColor.imsRed
```

## **TrueTypeMarkerSymbol.Overlap property**

### **Description**

The TrueTypeMarkerSymbol.Overlap property returns or sets a Boolean value indicating whether or not labels will overlap a TrueTypeMarkerSymbol.

### **Syntax**

TrueTypeMarkerSymbol.Overlap

### **Arguments**

None

### **Returned Value**

Boolean      True indicates that labels will be allowed to overlap the TrueTypeMarkerSymbol. False indicates that labels will not overlap the symbol. The default value is True.

### **Example**

```
Dim mTrueTypeMarkerSymbol  
Set mTrueTypeMarkerSymbol = Server.CreateObject("aims.TrueTypeMarkerSymbol")  
mTrueTypeMarkerSymbol.Overlap = False
```

## **TrueTypeMarkerSymbol.Shadow property**

### **Description**

The TrueTypeMarkerSymbol.Shadow property returns or sets a color constant for the shadow color of a TrueTypeMarkerSymbol.

### **Syntax**

TrueTypeMarkerSymbol.Shadow

### **Arguments**

None

### **Returned Value**

imsColor      Shadow color constant. The default value is imsBlack (0).

### **Example**

```
Dim mTrueTypeMarkerSymbol  
Set mTrueTypeMarkerSymbol = Server.CreateObject("aims.TrueTypeMarkerSymbol")  
mTrueTypeMarkerSymbol.Shadow = imsColor.imsBlack
```

## **TrueTypeMarkerSymbol.Transparency property**

### **Description**

The TrueTypeMarkerSymbol.Transparency property returns or sets the transparency coefficient for the TrueTypeMarkerSymbol. Lower numbers will display the symbol with greater transparency.

### **Syntax**

TrueTypeMarkerSymbol.Transparency

### **Arguments**

None

### **Returned Value**

Double Transparency coefficient. The default value is 1.0.

### **Example**

```
Dim mTrueTypeMarkerSymbol  
Set mTrueTypeMarkerSymbol = Server.CreateObject("aims.TrueTypeMarkerSymbol")  
mTrueTypeMarkerSymbol.Transparency = 1.0
```

## **ValueMapLabelRenderer.Add method**

### **Description**

The ValueMapLabelRenderer.Add method adds a symbol representing a value break to the ValueMapLabelRenderer.

### **Syntax**

ValueMapLabelRenderer.Add (Symbol, Range, Value, Upper, Lower)

### **Arguments**

Symbol	Symbol object to add.
Range	Type constant of the range (Exact, Range or Other)
Value	Variant value for exact symbol.
Upper	Upper variant value for range symbol.
Lower	Lower variant value for range symbol.

### **Returned Value**

Boolean      True indicates that the Add was successful. False indicates that an error occurred.

### **Example**

```
Dim mValueMapLabelRenderer  
Dim mTextSymbol  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mTextSymbol.FontColor = imsColor.imsRed  
mResult = mValueMapLabelRenderer.Add( mTextSymbol, imsRange.imsExact, 25, 0, 0 )  
mTextSymbol.FontColor = imsColor.imsBlue  
mResult = mValueMapLabelRenderer.Add( mTextSymbol, imsRange.imsRange, 0, 26, 31 )  
mTextSymbol.FontColor = imsColor.imsGreen  
mResult = mValueMapLabelRenderer.Add( mTextSymbol, imsRange.imsOther, 0, 0, 0 )
```

### **See Also**

[ShieldSymbol](#)  
[TextSymbol](#)  
[CalloutMarkerSymbol](#)  
[RasterShieldSymbol](#)

## **ValueMapLabelRenderer.Clear method**

### **Description**

The ValueMapLabelRenderer.Clear method clears all of the class break symbols and values from the ValueMapLabelRenderer.

### **Syntax**

ValueMapLabelRenderer.Clear()

### **Arguments**

None

### **Returned Value**

None

### **Example**

```
Dim mValueMapLabelRenderer  
Dim mTextSymbol  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mResult = mValueMapLabelRenderer.Add( mTextSymbol, imsRange.imsExact, 233, 0, 0)  
mValueMapLabelRenderer.Clear()
```

## **ValueMapLabelRenderer.Clone method**

### **Description**

The ValueMapLabelRenderer.Clone method clones the ValueMapLabelRenderer.

### **Syntax**

ValueMapLabelRenderer.Clone()

### **Arguments**

None

### **Returned Value**

ValueMapLabelRenderer      Cloned ValueMapLabelRenderer.

### **Example**

```
Dim mClone  
Dim mValueMapLabelRenderer  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
Set mClone = mValueMapLabelRenderer.Clone()
```

## **ValueMapLabelRenderer.Count property**

### **Description**

The ValueMapLabelRenderer.Count property returns the number of symbols in the ValueMapLabelRenderer.

### **Syntax**

ValueMapLabelRenderer.Count

### **Arguments**

None

### **Returned Value**

Long            Number of symbols.

### **Example**

```
Dim mValueMapLabelRenderer  
Dim mTextSymbol  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mResult = mValueMapLabelRenderer.Add( mTextSymbol, imsRange.imsExact, 233, 0, 0)  
mCount = mValueMapLabelRenderer.Count
```

## **ValueMapLabelRenderer.FWeight property**

### **Description**

The ValueMapLabelRenderer.FWeight property returns or sets a constant value indicating whether or not to consider the importance of features while labeling (feature weighting). This determines how important the feature is to the placement algorithm. The possible values are:

0 = imsNo\_weight

1 = imsMed\_weight

2 = imsHigh\_weight

If imsNo\_weight is specified, the feature has no importance and will be labeled. A value of imsHigh\_weight indicates that the feature is important and will not be labeled.

### **Syntax**

ValueMapLabelRenderer.FWeight

### **Arguments**

None

### **Returned Value**

imsWeight      Feature weight. The default value is imsNo\_weight

### **Example**

```
Dim mValueMapLabelRenderer  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
mValueMapLabelRenderer.FWeight = imsWeight.imsNo_weight
```

### **See Also**

imsWeight

## **ValueMapLabelRenderer.HTMLLabels property**

### **Description**

The ValueMapLabelRenderer.HTMLLabels property returns or sets a constant value indicating how many labels are to be drawn while labeling features. The possible values are:

- 0 = imsOne\_label\_per\_name
- 1 = imsOne\_label\_per\_shape
- 2 = imsOne\_label\_per\_part

### **Syntax**

ValueMapLabelRenderer.HTMLLabels

### **Arguments**

None

### **Returned Value**

imsHMLLabels Constant that determines how many labels are to be drawn to label a feature. The default value is imsOne\_label\_per\_name (0).

### **Example**

```
Dim mSValueMapLabelRenderer  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
mValueMapLabelRenderer.HTMLLabels = imsHMLLabels.imsOne_label_per_name
```

### **See Also**

imsHMLLabels

## **ValueMapLabelRenderer.LabelField property**

### **Description**

The ValueMapLabelRenderer.LabelField property returns or sets the name of the field containing text for labeling features.

### **Syntax**

ValueMapLabelRenderer.LabelField

### **Arguments**

None

### **Returned Value**

String      Name of the field.

### **Example**

```
Dim mValueMapLabelRenderer  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
mValueMapLabelRenderer.LabelField = "CNTRY_NAME"
```

## **ValueMapLabelRenderer.LabelPriorities property**

### **Description**

The ValueMapLabelRenderer.LabelPriorities property returns or sets a string value that determines where to place the label around the point. There are 8 positions for labels to be placed around a point. The LabelPriorities string contains 8 comma-delimited priorities for each position. A 0 at any position tells the placement algorithm not to place a label there at all. A value of 1 tells the algorithm to try and place a label at that position first.

### **Syntax**

ValueMapLabelRenderer.LabelProperties

### **Arguments**

None

### **Returned Value**

String            String value that tells the placement algorithm where to place the label around the point.

### **Example**

```
Dim mValueMapLabelRenderer  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
mValueMapLabelRenderer.LabelPriorities = "2,2,1,4,5,3,2,4"
```

## **ValueMapLabelRenderer.LLPosition property**

### **Description**

The ValueMapLabelRenderer.LLPosition property returns or sets a constant value that determines where to place a label on a line.

### **Syntax**

ValueMapLabelRenderer.LLPosition

### **Arguments**

None

### **Returned Value**

imsLLPosition A constant that determines where on the line to place a label. The default is imsPlaceAbove.

### **Example**

```
Dim mValueMapLabelRenderer  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
mValueMapLabelRenderer.LLPosition = imsLLPosition.imsPlaceNone
```

### **See Also**

imsLLPosition

## **ValueMapLabelRenderer.LookupField property**

### **Description**

The ValueMapLabelRenderer.LookupField property returns or sets the name of the field used for specifying ranges for RANGE or exact values for EXACT.

### **Syntax**

ValueMapLabelRenderer.LookupField

### **Arguments**

None

### **Returned Value**

String      Name of the field.

### **Example**

```
Dim mValueMapLabelRenderer  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
mValueMapLabelRenderer.LookupField = "CNTRY_NAME"
```

## **ValueMapLabelRenderer.Lower property**

### **Description**

The ValueMapLabelRenderer.Lower property returns or sets the lower value of a range for a symbol in a ValueMapLabelRenderer.

### **Syntax**

ValueMapLabelRenderer.Lower(index)

### **Arguments**

index            A long data type that is the index reference to the symbol.

### **Returned Value**

Variant            The value of the lower part of a range in a symbol.

### **Example**

```
Dim mValueMapLabelRenderer  
Dim mTextSymbol  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mResult = mValueMapLabelRenderer.Add( mTextSymbol, imsRange.imsRange, 0, 10, 31)  
mLower = mValueMapLabelRenderer.Lower(1)
```

## **ValueMapLabelRenderer.LWeight property**

### **Description**

The ValueMapLabelRenderer.LWeight property returns or sets a value indicating the importance of labels for a feature. If the labels are more important than the features themselves, keep Lweight set to HIGH. Possible values are:

- 0 = imsNo\_weight
- 1 = imsMed\_weight
- 2 = imsHigh\_weight

### **Syntax**

ValueMapLabelRenderer.LWeight

### **Arguments**

None

### **Returned Value**

imsWeight      Label weight. The default value is imsNo\_weight

### **Example**

```
Dim mValueMapLabelRenderer  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
mValueMapLabelRenderer.LWeight = imsWeight.imsNo_weight
```

### **See Also**

imsWeight

## **ValueMapLabelRenderer.Range** property

### **Description**

The ValueMapLabelRenderer.Range property returns or sets the type of the range (Exact, Range or Other) for a specified symbol in a ValueMapLabelRenderer. The possible values are:

0 = imsExact

1 = imsRange

2 = imsOther

### **Syntax**

ValueMapRenderer.Range(index)

### **Arguments**

index            A long data type. This is the index to a ValueMapLabelRenderer symbol.

### **Returned Value**

imsRange        Type of the range.

### **Example**

```
Dim mValueMapLabelRenderer  
Dim mTextSymbol  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mResult = mValueMapLabelRenderer.Add( mTextSymbol, imsRange.imsExact, 233, 0, 0)  
mRange = mValueMapLabelRenderer.Range(1)
```

### **See Also**

imsRange

## **ValueMapLabelRenderer.Remove method**

### **Description**

The ValueMapLabelRenderer.Remove method removes symbols from the ValueMapLabelRenderer by index.

### **Syntax**

```
ValueMapLabelRenderer.Remove(index)
```

### **Arguments**

index            A long index referencing the symbol.

### **Returned Value**

Boolean        True is returned if removal is successful. False is returned if an error has occurred (index out-of-bounds for example).

### **Example**

```
Dim mValueMapLabelRenderer  
Dim mTextSymbol  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mResult = mValueMapLabelRenderer.Add( mTextSymbol, imsRange.imsExact, 233, 0, 0)  
mResult = mValueMapLabelRenderer.Remove(1)
```

## **ValueMapLabelRenderer.RotationalAngles property**

### **Description**

The ValueMapLabelRenderer.RotationalAngles property returns or sets the possible angles that the label can be placed at relative to the labeled point.

### **Syntax**

ValueMapLabelRenderer.RotationalAngles

### **Arguments**

None

### **Returned Value**

Long            The rotational angle.

### **Example**

```
Dim mValueMapLabelRenderer  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
mValueMapLabelRenderer.RotationalAngles = 30
```

## **ValueMapLabelRenderer.Symbol property**

### **Description**

The ValueMapLabelRenderer.Symbol property returns or sets a symbol for the ValueMapLabelRenderer at the specified index.

### **Syntax**

`ValueMapLabelRenderer.Symbol( index )`

### **Arguments**

`index` Index of the symbol.

### **Returned Value**

`Object` Requested symbol.

### **Example**

```
Dim mValueMapLabelRenderer  
Dim mTextSymbol  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mResult = mValueMapLabelRenderer.Add( mTextSymbol, imsRange.imsExact, 233, 0, 0)  
mSymbol = mValueMapLabelRenderer.Symbol(1)
```

## **ValueMapLabelRenderer.Upper property**

### **Description**

The ValueMapLabelRenderer.Upper property returns or sets the upper value for a specified range in a ValueMapLabelRenderer.

### **Syntax**

ValueMapLabelRenderer.Upper( index )

### **Arguments**

index            Long indicating the index for the Range symbol.

### **Returned Value**

Object            The upper value of the specified Range.

### **See Also**

[ValueMapLabelRenderer.Lower](#)

### **Example**

```
Dim mValueMapLabelRenderer  
Dim mTextSymbol  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mResult = mValueMapLabelRenderer.Add( mTextSymbol, imsRange.imsRange, 0, 31, 76)  
mUpper = mValueMapLabelRenderer.Upper(1)
```

## **ValueMapLabelRenderer.Value** property

### **Description**

The ValueMapLabelRenderer.Value property returns or sets a value for exact symbol by index.

### **Syntax**

`ValueMapLabelRenderer.Value( index )`

### **Arguments**

`index`      A long index referencing a symbol.

### **Returned Value**

`Variant`      Value for exact symbol.

### **Example**

```
Dim mValueMapLabelRenderer  
Dim mTextSymbol  
Set mValueMapLabelRenderer = Server.CreateObject("aims.ValueMapLabelRenderer")  
Set mTextSymbol = Server.CreateObject("aims.TextSymbol")  
mResult = mValueMapLabelRenderer.Add( mTextSymbol, imsRange.imsExact, 233, 0, 0)  
mValue = mValueMapLabelRenderer.Value(1)
```

## ValueMapRenderer.Add method

### Description

The ValueMapRenderer.Add method adds a symbol to the ValueMapRenderer.

### Syntax

```
ValueMapRenderer.Add(symbol, range, value, upper, lower)
```

### Arguments

symbol	Symbol to add.
range	Type of the range (imsExact, imsRange or imsOther).
value	Value for exact symbol. Use 0 for types imsRange or imsOther.
upper	Upper value for range symbol. Use 0 for types imsExact or imsOther.
lower	Lower value for range symbol. Use 0 for types imsExact or imsOther.

### Returned Value

Boolean	True indicates that symbol is added successfully. False indicates that an error has occurred.
---------	---

### Example

```
Dim mValueMapRenderer  
Dim mSimpleMarkerSymbol  
Set mValueMapRenderer = Server.CreateObject("aims.ValueMapRenderer")  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.SimpleMarkerSymbol")  
mSimpleMarkerSymbol.Color = imsColor.imsRed  
mResult = mValueMapRenderer.Add( mSimpleMarkerSymbol, imsRange.imsExact, 25, 0, 0 )  
mSimpleMarkerSymbol.Color = imsColor.imsBlue  
mResult = mValueMapRenderer.Add( mSimpleMarkerSymbol, imsRange.imsRange, 0, 26, 31 )  
mSimpleMarkerSymbol.Color = imsColor.imsGreen  
mResult = mValueMapRenderer.Add( mSimpleMarkerSymbol, imsRange.imsOther, 0, 0, 0 )
```

### See Also

[TrueTypeMarkerSymbol](#)  
[SimpleMarkerSymbol](#)  
[RasterMarkerSymbol](#)  
[HashLineSymbol](#)  
[SimpleLineSymbol](#)  
[RasterFillSymbol](#)  
[SimplePolygonSymbol](#)  
[GradientFillSymbol](#)

## **ValueMapRenderer.Clear method**

### **Description**

The ValueMapRenderer.Clear method clears all of the class break symbols and values from the ValueMapRenderer.

### **Syntax**

```
ValueMapRenderer.Clear()
```

### **Arguments**

None

### **Returned Value**

None

### **Example**

```
Dim mValueMapRenderer  
Dim mSimpleMarkerSymbol  
Set mValueMapRenderer = Server.CreateObject("aims.ValueMapRenderer")  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.SimpleMarkerSymbol")  
mResult = mValueMapRenderer.Add( mSimpleMarkerSymbol, imsRange.imsExact, 266, 0, 0 )  
Set mValueMapRenderer.Clear()
```

## **ValueMapRenderer.Clone method**

### **Description**

The ValueMapRenderer.Clone method clones the ValueMapRenderer.

### **Syntax**

ValueMapRenderer.Clone()

### **Arguments**

None

### **Returned Value**

ValueMapRenderer      Cloned ValueMapRenderer.

### **Example**

```
Dim mClone  
Dim mValueMapRenderer  
Set mValueMapRenderer = Server.CreateObject("aims.ValueMapRenderer")  
Set mClone = mValueMapRenderer.Clone()
```

## **ValueMapRenderer.Count property**

### **Description**

The ValueMapRenderer.Count property returns the number of symbols in the ValueMapRenderer.

### **Syntax**

ValueMapRenderer.Count

### **Arguments**

None

### **Returned Value**

Long            Number of symbols.

### **Example**

```
Dim mValueMapRenderer  
Dim mSimpleMarkerSymbol  
Set mValueMapRenderer = Server.CreateObject("aims.ValueMapRenderer")  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.SimpleMarkerSymbol")  
mResult = mValueMapRenderer.Add(mSimpleMarkerSymbol, imsRange.imsExact, 233, 0, 0)  
mCount = mValueMapRenderer.Count
```

## **ValueMapRenderer.LookupField property**

### **Description**

The ValueMapRenderer.LookupField property returns or sets the name of the field used for specifying ranges for RANGE or exact values for EXACT.

### **Syntax**

ValueMapRenderer.LookupField

### **Arguments**

None

### **Returned Value**

String      Name of the field.

### **Example**

```
Dim mValueMapRenderer  
Set mValueMapRenderer = Server.CreateObject("aims.ValueMapRenderer")  
mValueMapRenderer.LookupField = "CNTRY_NAME"
```

## **ValueMapRenderer.Lower property**

### **Description**

The ValueMapRenderer.Lower property returns or sets the lower value of a range for a symbol in a ValueMapRenderer.

### **Syntax**

ValueMapRenderer.Lower(index)

### **Arguments**

index            A long data type that is the index reference to the symbol.

### **Returned Value**

Variant            The value of the lower part of a range in a symbol.

### **Example**

```
Dim mValueMapRenderer  
Dim mSimpleMarkerSymbol  
Set mValueMapRenderer = Server.CreateObject("aims.ValueMapRenderer")  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.SimpleMarkerSymbol")  
mResult = mValueMapRenderer.Add(mSimpleMarkerSymbol, imsRange.imsExact, 233, 0, 0)  
mLower = mValueMapRenderer.Lower(1)
```

## **ValueMapRenderer.Range property**

### **Description**

The ValueMapRenderer.Range property returns or sets the type of the range (Exact, Range or Other) for a specified symbol in a ValueMapRenderer. The possible values are:

0 = imsExact

1 = imsRange

2 = imsOther

### **Syntax**

`ValueMapRenderer.Range(index)`

### **Arguments**

`index`      A long data type. This is the index to a ValueMapRenderer symbol.

### **Returned Value**

`imsRange`      Type of the range.

### **Example**

```
Dim mValueMapRenderer  
Dim mSimpleMarkerSymbol  
Set mValueMapRenderer = Server.CreateObject("aims.ValueMapRenderer")  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.TextSymbol")  
mResult = mValueMapRenderer.Add( mSimpleMarkerSymbol, imsRange.imsExact, 233, 0, 0)  
mRange = mValueMapRenderer.Range(1)
```

### **See Also**

`imsRange`

## **ValueMapRenderer.Remove method**

### **Description**

The ValueMapRenderer.Remove method removes symbols from the ValueMapRenderer by index.

### **Syntax**

ValueMapRenderer.Remove(index)

### **Arguments**

index            A long index referencing the symbol.

### **Returned Value**

Boolean        True is returned if removal is successful. False is returned if an error has occurred (index out-of-bounds for example).

### **Example**

```
Dim mValueMapRenderer  
Dim mSimpleMarkerSymbol  
Set mValueMapRenderer = Server.CreateObject("aims.ValueMapRenderer")  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.TextSymbol")  
mResult = mValueMapRenderer.Add( mSimpleMarkerSymbol, imsRange.imsExact, 233, 0, 0)  
mResult = mValueMapRenderer.Remove(1)
```

## **ValueMapRenderer.Symbol property**

### **Description**

The ValueMapRenderer.Symbol property returns or sets a symbol for the ValueMapRenderer at the specified index.

### **Syntax**

ValueMapRenderer.Symbol( index )

### **Arguments**

index            Index of the symbol.

### **Returned Value**

Object            Requested symbol.

### **Example**

```
Dim mValueMapRenderer  
Dim mSimpleMarkerSymbol  
Set mValueMapRenderer = Server.CreateObject("aims.ValueMapRenderer")  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.TextSymbol")  
mResult = mValueMapRenderer.Add( mSimpleMarkerSymbol, imsRange.imsExact, 233, 0, 0)  
mSymbol = mValueMapRenderer.Symbol(1)
```

## **ValueMapRenderer.Upper property**

### **Description**

The ValueMapRenderer.Upper property returns or sets the upper value for a specified range in a ValueMapRenderer.

### **Syntax**

`ValueMapRenderer.Upper( index )`

### **Arguments**

`index` Long indicating the index for the Range symbol.

### **Returned Value**

`Object` The upper value of the specified Range.

### **See Also**

`ValueMapRenderer.Lower`

### **Example**

```
Dim mValueMapRenderer  
Dim mSimpleMarkerSymbol  
Set mValueMapRenderer = Server.CreateObject("aims.ValueMapRenderer")  
Set mSimpleMarkerSymbol = Server.CreateObject("aims.TextSymbol")  
mResult = mValueMapRenderer.Add( mSimpleMarkerSymbol, imsRange.imsRange, 0, 31, 76)  
mUpper = mValueMapRenderer.Upper(1)
```

## **ValueMapRenderer.Value property**

### **Description**

The ValueMapRenderer.Value property returns or sets a value for exact symbol by index.

### **Syntax**

`ValueMapRenderer.Value( index )`

### **Arguments**

`index` A long index referencing a symbol.

### **Returned Value**

`Variant` Value for exact symbol.

### **Example**

```
Dim mValueMapRenderer
Dim mSimpleMarkerSymbol
Set mValueMapRenderer = Server.CreateObject("aims.ValueMapRenderer")
Set mSimpleMarkerSymbol = Server.CreateObject("aims.SimpleMarkerSymbol")
mSimpleMarkerSymbol.Width = 10
mResult = mValueMapRenderer.Add( mSimpleMarkerSymbol, imsRange.imsExact, 266, 0, 0 )
mValue = mValueMapRenderer.Value(1)
```