ArcIMS® 9

Customizing ArcIMS®—Metadata Explorer

ESRI ArcGIS™

GIS by ESRI™

Contents

Introducing Metadata Explorer

1

IN THIS CHAPTER

- **What is Metadata Explorer?**
- **Metadata Explorer file organization**
- **Metadata Explorer UI organization**

ESRI® ArcIMS® software provides a suite of tools, allowing you to create effective Web sites for your mapping and geographic information system (GIS) needs. Metadata Explorer is a Web-based application that allows you to search and browse the contents of a Metadata Service—a central repository for metadata. Use Metadata Explorer to find data based on geographic extent, content type, content theme, or keyword or browse through the contents of a Metadata Service. Use the metadata returned by your search to determine whether the data it describes fits your needs.

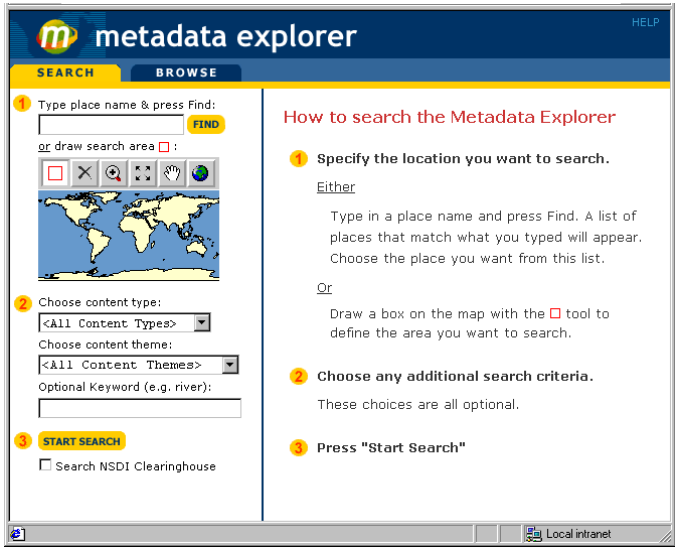
Customizing ArcIMS is a series of books that describes customizing the ArcIMS clients—HTML Viewer, Java™ Viewers, and Metadata Explorer—and creating viewers supported by the ActiveX®, ColdFusion®, and Java Connectors.

This book explains the foundation for customizing Metadata Explorer. For Metadata Explorer installation and configuration instructions, see *ArcIMS Installation Guide*. For details on using Metadata Explorer to search or browse for metadata, see C*reating and Using Metadata Services*.

This book assumes that you have a working knowledge of HTML and familiarity with JavaScript™.

What is Metadata Explorer?

Metadata Explorer is a Web-based application that can be used to find data based on geographic extent, content type, content theme, or keyword or to browse through the contents of a Metadata Service.



Metadata Explorer utilizes JavaServer Pages™ (JSP)—HTML pages with Java functionality within them. This enables the major work of the application to run server-side, taking the load off the client machine. In addition to this core set of JSP files, a number of JavaScript files provide much of the interactive functionality present in Metadata Explorer.

Metadata Explorer is an open source application that can be easily customized to the specific needs of your organization.

You can choose to alter any or all of the following:

- Logos, color scheme, and other associated application graphics.
- Application name displayed.
- Help documentation.
- Search criteria interface.
- Result views (summary and detail).
- Map display.

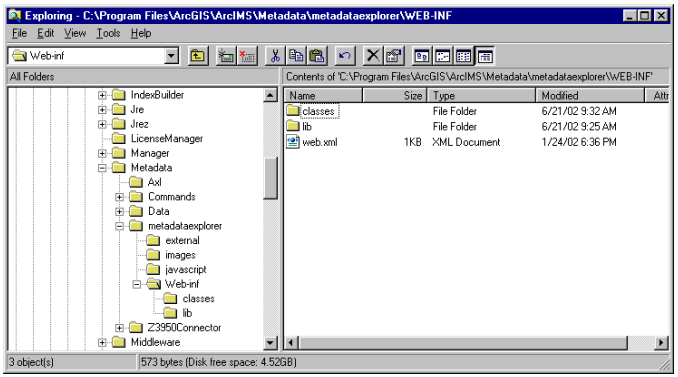
Additionally, parameters within the configuration files of Metadata Explorer govern further customizations including the ability to:

- Restrict access to Metadata Explorer via a username and password.
- Set the number of items per page that appear in a results display.
- Choose the Metadata Service associated with Metadata Explorer.
- Employ the gazetteer functionality.

Metadata Explorer file organization

Directory structure

The installed Metadata Explorer directory (<ArcIMS Directory>/ArcIMS/Metadata/metadataexplorer) contains a number of JSP (.jsp) and HTML (.html) files, along with four subdirectories—external, images, javascript, and WEB-INF.



The **external** subdirectory contains the metadata.css file—a cascading stylesheet defining the appearance of text within Metadata Explorer such as font face, size of text, color of text, and format of hyperlinks.

Graphic images used in the pages of Metadata Explorer, such as buttons, icons, logos, and headers, are included in the **images** subdirectory. Customization may include replacing these existing images to create your own corporate or departmental look.

The **javascript** subdirectory contains the JavaScript library for Metadata Explorer. These JavaScript files contain the functions that perform many of the common operations for Metadata Explorer.

The **WEB-INF** subdirectory contains the web.xml file and is further divided into lib and classes directories:

- The **lib** directory contains the Java Archive (JAR) files—packaged Java class files that enable communication with the Metadata Server component of the ArcIMS Spatial Server. This directory also contains the Tag Library Descriptor (TLD) files that define the custom JSP tags used.

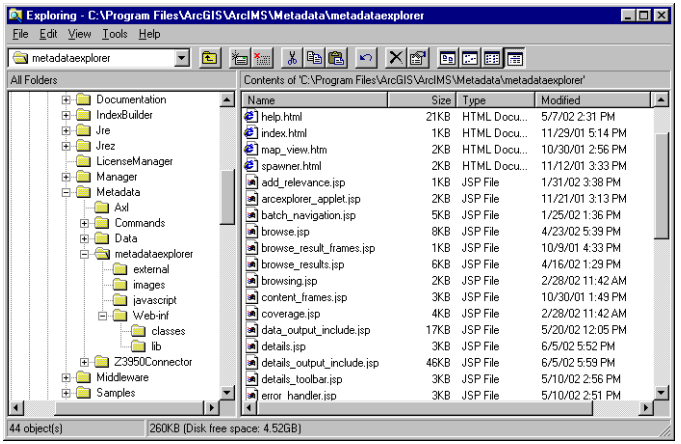
- The property files of Metadata Explorer are included in the **classes** directory. In addition, this directory also contains a sample Access Control List (ACL) file, aimsacl.xml, which can be used to set up username and password restrictions on the associated Metadata Service.

JSP and HTML files

There are approximately 40 files—four HTML and the rest JSP—that define the page content, layout, and user interface for Metadata Explorer.

They are located in the /metadataexplorer directory.

A discussion of each of these files, including the function they serve, what other files call them, and any other information that may be useful when customizing Metadata Explorer for your purposes, follows. The files are listed in alphabetical order.



File name	Purpose	Called by	Additional information
arcexplorer_applet.jsp	Retrieves the Geography Network℠ Document (GND) file for a dataset when Add to ArcExplorer™ is selected.	navigation.js	This file is used exclusively when Metadata Explorer is launched from within ArcExplorer. The Metadata Service initially creates the GND document when metadata documents that describe ArcIMS Image or Web Map Server (WMS) Services are published.

File name	Purpose	Called by	Additional information
batch_navigation.jsp	Provides navigation between batches of results in a display.	browse_results.jsp data_output_include.jsp search_results.jsp	The batch size—or the number of results appearing on a page at a time—is defined in the aimsmeta.properties file. If the number of results is greater than the batch number specified, the results are shown in batches and navigation tools between batches are present at the top and bottom of each page. Use the navigation tools to jump to the first page, the last page, or to a specific page of results within a three-page range of the current page.
browse.jsp	Displays the browsing path and the child folders of the current folder during browsing.	content_frames.jsp *(if the goTo parameter is set to browse)*	This is loaded into the left-hand frame of a frameset. The right-hand frame is browse_result_frames.jsp. After initializing a series of JavaScript functions, which enable navigation through different folders while browsing, the remaining code in browse.jsp writes out the path of navigation and, below it, the child folders of the one currently being viewed. If the current browse location is the home/root directory, the home folder's datasets are written out in the right-hand pane instead.
browse_result_frames.jsp	Sets up a frameset that holds the toolbar (toolbar.jsp) and browse results (browsing.jsp).	content_frames.jsp *(if the goTo parameter is set to browse)*	This is loaded into the right-hand pane of a frameset. The left-hand pane is browse.jsp.

File name	Purpose	Called by	Additional information
browse_results.jsp	Retrieves and displays browse results.	browsing.jsp	A browse request is sent by browse_results.jsp to the Metadata Service for a list of all the datasets that belong to the current folder. Once a response is received, the code from batch_navigation.jsp is called to write out the batch navigation tools. Finally, data_output_include.jsp is called to write out the results.
browsing.jsp	Displays an animated image (BROWSING) while the results of a browse request are being retrieved and processed.	browse_result_frames.jsp	When browsing.jsp loads, it immediately calls a JavaScript function to change the page location to browse_results.jsp. The new page cannot be displayed until all content is available. While the results are being generated in browse_results.jsp, browsing.jsp displays an animated BROWSING image. Once the results are ready to be displayed, the page location switches from browsing.jsp to browse_results.jsp.
content_frames.jsp	Sets up a dynamic frameset and calls files to satisfy action requests such as searching, browsing, viewing dataset details, viewing a map, or using the gazetteer.	explorer.jsp navigation.jsp tabs.jsp	Initially called by explorer.jsp, content_frames.jsp is the entry point to Metadata Explorer. It is also called anytime the user requests a different action, for instance, switching from browsing to searching or to a details or map view for any particular dataset. In these cases, content_frames.jsp is called by tabs.jsp or navigation.js.

File name	Purpose	Called by	Additional information
coverage.jsp	Gives a graphical representation of the coverage area that a dataset represents.	details.jsp (JavaScript function, highlight)	When View Coverage Area on the details page is clicked, coverage.jsp opens a new browser window and shows a green box around the coverage area of the dataset.
data_output_include.jsp	Writes out the results from a browse or search request.	browse_results.jsp print.jsp search_results.jsp	This file first uses the JSP iterateContentType tag to write out the current page's result summary—a hyperlinked list showing the number of datasets of each content type in the current batch of displayed results. It then uses the JSP iterateMetadata tag to loop through each of the datasets in a result set, writing out information about each one and providing links to view details; a map, if available; or download the data, if possible. Finally, batch_navigation.jsp is called to write out the batch navigation tools at the bottom of the result display.
details.jsp	Sets up JavaScript functions used for displaying the coverage area of a dataset.	content_frames.jsp *(if the goTo parameter is set to details)*	Loaded as part of a frameset, details.jsp appears in the bottom row of two. The other row loads details_toolbar.jsp. After setting up JavaScript functions used to display the coverage area of a dataset and variables needed for details display, details.jsp calls details_output_include.jsp to write out the details of a dataset.

File name	Purpose	Called by	Additional information
details_output_include.jsp	Displays an extensive view of a dataset's metadata.	details.jsp print.jsp	Using the docID of the metadata being viewed in conjunction with the getMetadataDocument JSP tag, details_output_include.jsp extracts the value of desired metadata elements using the loadElements and getElement JSP tags. Once the element values have been extracted and saved to a variable on the page, HTML is used to display the detailed metadata. Additionally, when applicable, buttons are drawn that allow the user to view a map of the dataset or to download it. Finally, a hyperlinked list of all datasets related to the one in question is written using the iterateMetadata tag.
details_toolbar.jsp	Writes out the dataset title and draws the print button in a details display.	content_frames.jsp *(if the goTo parameter is set to details)*	This page is loaded as part of a frameset, the top row of two. The other row loads details.jsp.
error_handler.jsp	Displays an error message when a JSP page throws a Java exception.	browse.jsp browse_results.jsp data_output_include.jsp details_output_include.jsp find_place.jsp login.jsp map_view.jsp print.jsp search_form.jsp search_map.jsp search_results.jsp	Any JSP page that has the potential to throw a Java exception during processing can call error_handler.jsp. If a Java exception error occurs, the remaining code on the page is not executed; instead, the page is redirected to error_handler.jsp. If the source page was the gazetteer or a map view, the error message displayed will be specific to the source page. Otherwise, a general message that an error has occurred is displayed.

File name	Purpose	Called by	Additional information
explorer.jsp	Serves as the entry point to Metadata Explorer by setting up global JavaScript functions and variables and creating the initial user interface (UI) frameset.	index.html login.jsp spawner.html title.jsp	If user authentication is enabled in Metadata Explorer property files, explorer.jsp redirects to login.jsp. Otherwise, global JavaScript variables and functions are defined and initialized and the UI frameset is created. The exact files loaded by explorer.jsp depend on how Metadata Explorer was launched. Possibilities include a new browser window from within an application such as ArcMap™, ArcExplorer—Java Edition; or from a viewer.
find_place.jsp	Displays gazetteer results.	find_place_frames.jsp	This file is loaded as part of a frameset, the bottom row of two. The other row loads find_place_header.jsp. First, find_place.jsp initializes a series of JavaScript functions necessary for the gazetteer search result display. Then, using the JSP search tag, a list of place names matching the user's input is retrieved. Finally, the results are written out as a hyperlinked list using the iterateMetadata tag. Each link updates the search map in the left-hand pane with the extent of the selected place and changes color once it has been visited.
find_place_frames.jsp	Sets up the frameset for the gazetteer search results.	content_frames.jsp *(if the goTo parameter is set to findPlace)*	This file is loaded as part of a frameset, into the right-hand pane. The left-hand pane is search_frames.jsp. Find_place_frames.jsp sets up a frameset to hold the gazetteer search result set header (find_place_header.jsp) and the gazetteer search results (find_place.jsp).

File name	Purpose	Called by	Additional information
find_place_header.jsp	Writes out a header message preceding gazetteer search results.	find_place_frames.jsp	Find_place_header.jsp is loaded as part of a frameset, the top row of two. The other row loads find_place.jsp. Find_place_header.jsp writes out a message indicating the status of a gazetteer search as well as user instructions for interacting with the results written out by find_place.jsp.
help.html	Displays user help for working with Metadata Explorer.	title.jsp	When the help link (top-right corner of the interface) is clicked, a new browser window opens with the contents of help.html. This file gives the user detailed instructions for using Metadata Explorer functions such as searching, browsing, viewing details, navigating through a map, and printing.
help_search.jsp	Displays initial instructions on how to search using Metadata Explorer.	search_result_frames.jsp	When Metadata Explorer is initiated or switched into search mode, the initial searching screen shows the help_search.jsp file in the right-hand pane of the frameset.
index.html	Redirects the browser to explorer.jsp.	None	In many Web applications, if the initial URL does not specify a file name but rather ends on a directory name, the default page, index.html, is loaded. In Metadata Explorer, index.html forwards the page location to explorer.jsp, the official entry point to the application.

File name	Purpose	Called by	Additional information
login.jsp	Provides a login dialog prior to accessing a restricted Metadata Explorer.	browse.jsp browse_results.jsp details_output_include.jsp explorer.jsp login.jsp map_view.jsp print.jsp search_results.jsp title.jsp	If user authentication is enabled, each JSP page making a connection to the Metadata Service must have username and password parameters defined. If these parameters are not defined, the user is required, via login.jsp, to input a username and password. Login parameters are then sent through to the Metadata Service using the validateConnection tag. If a successful login occurs, the login parameters are stored as session variables for eventual use, and the user is directed back to the page that initially redirected to login.jsp. Otherwise, the user is informed of the error and the login.jsp page is reloaded for another attempt at username and password validation.
map_authenticate.jsp	Provides a login dialog prior to displaying a restricted Image or WMS Service.	map_print.jsp map_view.jsp	When a user attempts to view or print an Image or WMS Service that requires authentication and login parameters for the service have not yet been defined, Metadata Explorer redirects to map_authenticate.jsp. Here, the user is presented with a login dialog to input a username and password. Upon submitting the login form, the user is sent back to either map_print.jsp or map_view.jsp (depending on which file called map_authenticate.jsp originally). If the login was successful, the map displays. Otherwise, the user is sent back to map_authenticate.jsp to try the login again.

File name	Purpose	Called by	Additional information
map_legend.jsp	Displays a map's legend.	content_frames.jsp *(if the goTo parameter is set to map)*	Map_legend.jsp is loaded as part of a frameset, the left-hand column of the bottom row. The right-hand column is the map itself (map_view.htm), and the top row is map_title.jsp. Initially, map_legend.jsp loads as a blank page—a placeholder for the legend. Once the map in the right-hand frame finishes loading, it makes a call to map_legend.jsp, replacing the legend placeholder with the legend image as a parameter to the page.
map_print.jsp	Displays a map in printable format.	map_authenticate.jsp map_general.js (JavaScript printIt function)	Map_print.jsp reprocesses the map_view.jsp code that displays the map and shows it on the map_print.jsp page. The print command in a Web browser menu can be used to output the image.
map_title.jsp	Writes out the map title.	content_frames.jsp *(if the goTo parameter is set to map)*	Map_title.jsp is loaded as part of a frameset, the top row of two. The other row is a frameset showing the map view and legend.
map_view.htm	Initializes a layer for displaying the map and displays an animated LOADING image while the map image is being retrieved from the Metadata Service.	content_frames.jsp *(if the goTo parameter is set to map)*	Map_view.htm is loaded as part of a frameset, the right-hand column of the bottom row. The left-hand column is the legend of the map (map_legend.jsp), and the top row is map_title.jsp. Once the map image is received from the Metadata Service, map_view.htm initializes parameters specifying the width and height of the layer, then makes a JavaScript call to change the location of the page to map_view.jsp, which displays the map itself.

File name	Purpose	Called by	Additional information
map_view.jsp	Displays the map image.	map_view.htm map_authentication.jsp	Map_view.jsp is called from either map_view.htm when the map is ready to be loaded or from map_authentication.jsp if the user was redirected to the login screen before the map could be displayed. Map_view.jsp begins by initializing a series of JSP variables such as the initial map extent, the parameters for a full map extent, the image height and width, and the map units. It then extracts information from the map's dataset such as the title of the map and its publisher. Finally, map_view.jsp loads the Image Service by creating a map object and generating the map image in a layer that may include acetate objects such as the scale bar and map text. Above the layer, a map toolbar with standard buttons, such as zoom, pan, and extent navigation, is also created. Map_view.jsp has a separate set of code statements for displaying WMS Image Services.
print.jsp	Prints out a list of results or a dataset's details.	details_toolbar.jsp toolbar.jsp	Print.jsp is called whenever the Print button is clicked from any Metadata Explorer state other than the map view. Print.jsp reprocesses the code for a search, browse, or details request, regenerating the results and displaying them in a printer-friendly format on the print.jsp page. This page can then be printed using the print command available in a standard browser menu.
search_form.jsp	Builds the interface components used to define a search query.	search_frames.jsp	Search_form.jsp is loaded as part of a frameset into the bottom row of two. The top row displays the search map and a gazetteer search box if gazetteer searching has been enabled.

File name	Purpose	Called by	Additional information
search_frames.jsp	Sets up a frameset for the search interface that holds the search map (search_map.jsp) and the search form (search_form.jsp).	content_frames.jsp *(if the goTo parameter is set to search, helpSearch, or findPlace)*	Content_frames.jsp is loaded as part of a frameset into the left-hand pane. The right-hand pane is search_result_frames.jsp or find_place_frames.jsp.
search_map.jsp	Displays a map in the search interface with tools to define a search extent.	find_place.jsp search_frames.jsp	Search_map.jsp first sets up a series of JSP variables defining the coordinates of the map's initial and full extent and the extent of a box on the map if one is to be drawn. It then creates the map object by first connecting to the map service, then retrieving the map image. In retrieving the map image, an acetate layer is created for all acetate objects that may need to be added to the map. If a bounding box is to be drawn, an acetate object is created for this box. If search_map.jsp was called from a gazetteer search (find_place.jsp), another acetate object is created for the place name of the chosen gazetteer result. Once all variables, objects, and layers have been created and initialized, search_map.jsp displays the gazetteer find box, if it is active; draws a toolbar for map controls such as zoom, pan, and extent navigation; and, finally, creates a layer for and displays the map and acetate objects.

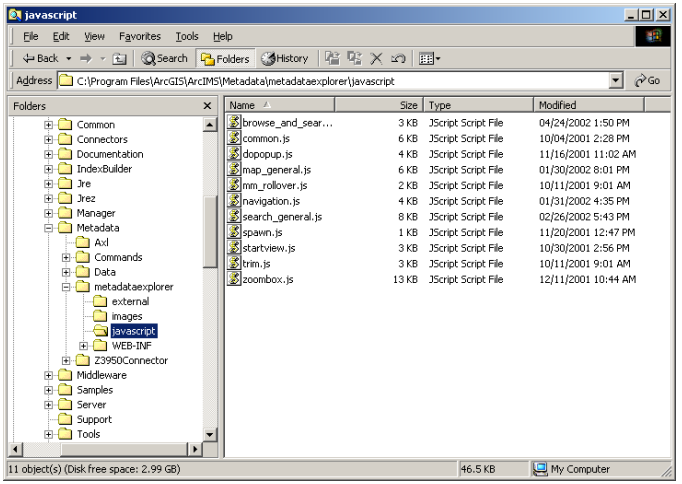
File name	Purpose	Called by	Additional information
search_result_frames.jsp	Sets up a frameset that holds either search instructions, or the toolbar (toolbar.jsp) and the search results (searching.jsp).	content_frames.jsp *(if the goTo parameter is set to search or helpSearch)*	Search_result_frames.jsp is loaded as part of a frameset into the right-hand pane. The left-hand pane is search_frames.jsp. Depending on the value of the goTo parameter passed into the page, search_result_frames.jsp sets up a dynamic frameset. If the goTo parameter is helpSearch, search_result_frames.jsp builds a frameset to display the help_search.jsp file. If the goTo parameter is search, search_result_frames.jsp builds a frameset to hold the toolbar (toolbar.jsp) and the search results (searching.jsp).
search_results.jsp	Retrieves and displays search results.	searching.jsp	Search_results.jsp first sends a search request to the Metadata Service for a list of all the datasets matching the criteria specified in the search interface. Once a response is received, the code from batch_navigation.jsp is called to write out the batch navigation tools. Finally, data_output_include.jsp is called to write out the results.
searching.jsp	Displays an animated SEARCHING image while the results of a search request are being retrieved and processed.	search_result_frames.jsp	When searching.jsp loads, it immediately calls a JavaScript function to change the page location to search_results.jsp. The new page cannot be displayed until all content is available. While the results are being generated in search_results.jsp, searching.jsp displays an animated SEARCHING image. Once the results are ready to be displayed, the page location switches from searching.jsp to search_results.jsp.

File name	Purpose	Called by	Additional information
spawner.html	Shows examples of launching Metadata Explorer into different states and from different applications.	None	Spawner.html does not play an active role in the current Metadata Explorer code. Instead, it illustrates how to integrate Metadata Explorer into a site or application by providing examples for launching Metadata Explorer into different states and from different applications. For more details refer to 'Integrating Metadata Explorer into a site' in Chapter 2.
tabs.jsp	Draws the navigational tabs near the top of the interface.	explorer.jsp navigation.js tabs.jsp	In addition to drawing the navigational tabs, tabs.jsp has a series of JavaScript functions that save the arguments of the last search or browse request executed. This is used when returning to a list of results from a details or map view—the browse or search request that generated the results is sent to the Metadata Service again and the results are retrieved anew. Since this file saves the search or browse request parameters, the user does not need to input all of the request criteria again.
title.jsp	Draws the title graphic and provides help and login/logout links at the top of the interface.	explorer.jsp	None
toolbar.jsp	Writes out the number of records	browse_and_search_result.js browse_result_frames.jsp search_result_frames.jsp	None

File name	Purpose	Called by	Additional information
	returned and displays a print button in a result set.		

JavaScript files

Metadata Explorer uses a library of JavaScript functions that are stored in files within the JavaScript subdirectory. These files are generally organized according to the JSP files that they are referenced in.



- **Map files** provide functionality for creating, drawing, and working with a map image.
- **Navigation files** provide functionality for navigating through results, result set batches, and different application modes.
- **Miscellaneous files** have common functions used throughout Metadata Explorer or are used for sample/test files that are included with the source code.

A discussion of each of these files, including what other files they are referenced in and any other information that may be useful, follows. The files are listed in alphabetical order.

Category	File name	Referenced in	Details
Map	common.js	map_view.jsp search_map.jsp	Functions in common.js provide code for creating and working with the map layer. It also performs a check to determine the browser and browser version being used, since different browsers and versions implement layering differently.

Category	File name	Referenced in	Details
Map	dopopup.js	map_view.jsp	It is used to open up a popup window when the user clicks the print button in a map view.
Map	map_general.js	map_view.jsp	It sets up the variables needed to work with a map of a dataset. These include map property variables such as name, owner, and units; map extent variables; map image and size variables; legend variables; and scale variables. This file also sets up the functions needed to work with a map, such as drawing a zoom box, zooming and panning, extent navigation, map refresh, and map printing.
Map	search_general.js	search_map.jsp	The variables needed to work with the map in the search interface or from gazetteer results are set up in search_general.js. These include map extent variables, map image and size variables, and scale variables. It also sets up the functions needed to work with the search map such as drawing/deleting an extent box, writing the place name from a gazetteer search, zooming and panning, and map refresh.
Map	startview.js	map_view.html	Used to create the layer that the map will be drawn in, startview.js also has a series of functions that determine the browser and browser version being used. These are necessary because different browsers and versions implement layering differently.
Map	zoombox.js	map_view.jsp search_map.jsp	It provides JavaScript variables and functions for zooming and panning a map. The zoom action can be triggered either by drawing a box or clicking a single point on the map.

Category	File name	Referenced in	Details
Navigation	browse_and_search_results.js	browse_results.jsp data_output_include.jsp search_results.jsp	The various functions in this file are used by the browse and search result pages to store the query parameters, reload the toolbar above the returned results so that it displays the number of records in the result set, and set an anchor when the details or map button is clicked so that the user is returned to that record instead of the top of the result set.
Navigation	navigation.js	browse_results.jsp details.jsp search_results.jsp tabs.jsp	A subset of the functions in this file is used by browse_results.jsp and search_results.jsp to navigate between different batches of results as well as to launch the arcexplorer_applet.jsp file when the Add to ArcExplorer or Add to ArcMap button is clicked for a result. Another subset of the functions in this file is used by details.jsp and tabs.jsp to load the appropriate tabs and content into the interface, overwriting the standard two-pane frameset with a single-frame interface.
Misc	mm_rollover.js	any file that uses image rollovers	Any file that uses image rollovers references this file, mm_rollover.js. The functions in this file are Macromedia Dreamweaver standard rollover functions.
Misc.	spawn.js	spawner.html	spawn.js is referenced by spawner.html, the test file used to experiment with different ways of launching Metadata Explorer. spawn.js has functions to open a new browser window with Metadata Explorer launched in a defined state.
Misc.	trim.js	spawner.html	trim.js is referenced by spawner.html, the test file used to experiment with different ways of launching Metadata Explorer. The functions in trim.js return a copy of the string passed into them without leading spaces at the front, back, or both sides of the string.

Other important files

A number of other files may play an important role in any customizations you wish to make to Metadata Explorer. The files, and their directory location, are listed in alphabetical order.

File name	Purpose	Called by	Additional information
aimsmeta.properties *Location*: WEB-INF\classes directory	Defines connection properties and interface variables.	browse.jsp* browse_results.jsp* coverage.jsp* details_output_include.jsp* explorer.jsp* find_place.jsp* help_search.jsp* login.jsp* map_view.jsp* print.jsp* search_frames.jsp* search_map.jsp* search_results.jsp* title.jsp* *<aims:getSettings> tag	Examples of connection properties defined in aimsmeta.properties are the names of the associated metadata, gazetteer, and searchmap services and the host machine and port on which each service runs. Examples of interface variables defined in aimsmeta.properties are the visibility of the gazetteer, the batch size of a result set, and whether user authentication is required for browsing/searching. If authentication is turned on (set to optional or required), properties in authenticate.properties will also need to be set to fully enable the user authentication process.
authenticate.properties *Location*: WEB-INF\classes directory	Defines authentication parameters if authentication is enabled.	This file is used by the ArcIMS Java Connector and is not called by any Metadata Explorer files.	Authentication allows restrictions to be placed on access to ArcIMS services. If login was set to optional or required in aimsmeta.properties, authenticate.properties has to be modified to fully enable authentication. In authenticate.properties, specify either the path to the file-based Access Control List or set the Java Database Connectivity (JDBC) authentication parameters.

File name	Purpose	Called by	Additional information
metadata.css *Location*: External directory	Defines text styles (font, size, color, weight) for textual components of the interface.	Any file that writes out text to the screen.	Text appearing throughout the interface can be assigned to a specific class. The style properties of each class are defined by metadata.css.
res.properties *Location*: WEB-INF\classes directory	Defines text strings used throughout Metadata Explorer.	Any file that writes out text to the screen.	Having a separate file that defines all the text strings used in Metadata Explorer simplifies customization of each string for a specific language or locale.
sample_aimsacl.xml *Location*: WEB-INF\classes directory	Provides an example of an ACL file.	None	An ACL file is used to set up usernames and passwords for the different access levels associated with Metadata Services. For more details on these permission levels, refer to *Creating and Using Metadata Services*.
web.xml *Location*: WEB-INF directory	Specifies the entry point (explorer.jsp) to Metadata Explorer and the path to the Tag Library Descriptor files, which define the custom tags used in JSP files.	This file is used by the Web server and is not called by any Metadata Explorer files.	None.

WEB-INF/lib directory

The /lib directory contains Java Archive files and Tag Library Definition files. JARs are packaged files that contain all of the Java class files needed by Metadata Explorer. TLD files define the custom JSP tags used throughout the JSP files of Metadata Explorer.

Image files

All of the graphics and tool images used throughout Metadata Explorer are located in the Images subdirectory. Customize the appearance of the application by replacing these graphics with your own and making the necessary modifications to the files that reference them. The files are listed in alphabetical order.

File name	Functional tool	Referenced in	Details
addarcexplorer_off.gif and **addexplorer_press.gif**		data_output_include.jsp details_output_include.jsp	If Metadata Explorer was launched from ArcExplorer and a particular dataset has a Geography Network Document property defined, the Add to Explorer rollover pair button appears in a result or details display.
addarcmap_off.gif and **addarcmap_press.gif**		data_output_include.jsp details_output_include.jsp	If Metadata Explorer was launched from ArcMap and a particular dataset has the GND property defined, the Add to ArcMap rollover pair button appears in a result or details display.
addarcreader_off.gif and **addarcreader_press.gif**		data_output_include.jsp details_output_include.jsp	If Metadata Explorer was launched from ArcReader™ and a paticular dataset has the GND property defined, the Add to ArcReader rollover pair button appears in a result or details display.
addmap_off.gif and **addmap_press.gif**		data_output_include.jsp details_output_include.jsp	If Metadata Explorer was launched from a viewer and the dataset has the GND property defined, the Add to Map rollover pair button appears in a result or details display.

File name	Functional tool	Referenced in	Details
browse_tab_off.gif and **browse_tab_select.gif**		tabs.jsp	Browse rollover pair navigational tab that provides a link to the browse mode.
browsing.gif		browsing.jsp	An animated image that is displayed while browse results are being retrieved.
close_button.gif		coverage.jsp map_print.jsp print.jsp	Displayed in any popup window that is opened by Metadata Explorer. Clicking on it closes the window.
details_tab_off.gif and **details_tab_select.gif**		tabs.jsp	Details rollover pair navigational tab that provides a link to the details when in map mode.
find_off.gif and **find_press.gif**		search_map.jsp	Find rollover pair button displayed in the search interface beside the gazetteer find box. When clicked, a gazetteer search is initiated.
loading.gif		map_view.jsp	An animated image that is displayed while a map is loading.
login_off.gif and **login_press.gif**		login.jsp map_authenticate.jsp	Login rollover pair button that initiates a login or authentication request when clicked.
map_tab_off.gif and **map_tab_select.gif**		tabs.jsp	Map rollover pair navigational tab that provides a link to the map when in details mode.
me_title.gif		title.jsp	The main title bar appearing in the top frame of the interface.

File name	Functional tool	Referenced in	Details
meheader_lg.gif		login.jsp map_print.jsp print.jsp	The title bar that appears on any popup window.
red_box.gif		help_search.jsp search_map.jsp	An image of a hollow box with a red border that indicates to the user which map tool button to select when defining an extent on the search map.
search_1.gif **search_2.gif** **search_3.gif**		search_form.jsp help_search.jsp	These images identify the sequential logic to defining search criteria.
search_off.gif and **search_press.gif**		search_form.jsp	Start Search rollover pair button displayed on the search form. When clicked, a search is initiated.
search_tab_off.gif and **search_tab_select.gif**		tabs.jsp	Search rollover pair navigational tab that provides a link to the search mode.
searching.gif		searching.jsp	An animated image that is displayed while search results are being retrieved.
toolcleararea_off.gif and **toolcleararea_press.gif**	Clear Area map tool	search_map.jsp	This tool clears a selected extent on a map.
toolextent_off.gif and **toolextent_press.gif**	Zoom to Full Extent map tool	map_view.jsp search_map.jsp	This tool reloads a map to full extent, removing any zoomed or panned views.

File name	Functional tool	Referenced in	Details
toolforwardextent_disabled.jpg	Forward Extent map tool placeholder	map_view.jsp	A forward extent is the next zoomed in extent from the one currently shown. It is only set if the user has previously gone to a zoomed in extent greater than the current view. If this extent has not yet been defined, toolforwardextent_disabled.jpg is displayed instead of the Forward Extent tool. It shows the Forward Extent tool in a disabled view and is not clickable.
toolforwardextent_off.gif and **toolforwardextent_press.gif**	Forward Extent map tool	map_view.jsp	A forward extent is the next zoomed in extent from the one currently shown. It is set only when the user has previously gone to a zoomed in extent greater than the current view. This tool sets the map to that next zoomed in extent.
toolpan_disabled.jpg	Pan map tool placeholder	map_view.jsp	This image is displayed instead of the Pan tool when the map is at full extent and cannot be panned. The image shows the Pan tool in a disabled view and is not clickable.
toolpan_off.gif and **toolpan_press.gif**	Pan map tool	map_view.jsp search_map.jsp javascript\map_general.js javascript\search_general.js	This tool enables the user to click on any point on the map and, while holding the mouse button down, drag the map around. Releasing the mouse button will cause the map to be reloaded with the new extent defined by the pan.

File name	Functional tool	Referenced in	Details
toolpreviousextent_disabled.jpg	Previous Extent map tool placeholder	map_view.jsp	A previous extent is the next zoomed out extent from the one currently shown. It is set only when the user has previously gone to a zoomed out extent greater than the current view. If this extent has not yet been defined, toolpreviousextent_disabled.jpg is displayed instead of the toolpreviousextent tool. It shows the Previous Extent tool in a disabled view and is not clickable.
toolpreviousextent_off.gif and **toolpreviousextent_press.gif**	Previous Extent map tool	map_view.jsp	A previous extent is the next zoomed out extent from the one currently shown. It is set only when the user has previously gone to a zoomed out extent greater than the current view. This tool sets the map to that next zoomed out extent.
toolprint_off.gif and **toolprint_press.gif**	Print tool	toolbar.jsp details_toolbar.jsp map_view.jsp	This tool launches a new browser window displaying the current results or map view in a printable format.
toolselectarea_off.gif and **toolselectarea_press.gif**	Select Area map tool	search_map.jsp javascript\search_general.js	This tool enables the user to click on any point on the map and, while holding the mouse button down, draw a box on the map. Releasing the mouse button will finalize the box drawn on the map.
toolzoomin_disabled.jpg	Zoom In map tool placeholder	None.	toolzoomin_disabled.jpg shows the Zoom In symbol in a disabled view. It is currently not used in Metadata Explorer but is available for a customized application.

File name	Functional tool	Referenced in	Details
toolzoomin_off.gif and **toolzoomin_press.gif**	Zoom In map tool	map_view.jsp search_map.jsp javascript\map_general.js javascript\search_general.js	This tool enables the user to click on any point on the map and, while holding the mouse button down, draw a box on the map. Releasing the mouse button will cause the map to be reloaded, zoomed in to the extent of the box that was drawn.
toolzoomout.gif	Zoom Out map tool for the search map	search_map.jsp	This tool zooms the map out by a defined interval.
toolzoomout_disabled.jpg	Zoom Out map tool placeholder	map_view.jsp	When the map is at full extent and cannot be zoomed out further, toolzoomout_disabled.jpg is displayed instead of the Zoom Out tool button. The button shows the Zoom Out symbol in a disabled view and is not clickable.
toolzoomout_off.gif and **toolzoomout_press.gif**	Zoom Out map tool	map_view.jsp javascript\map_general.js	This tool enables the user to click on any point on the map and, while holding the mouse button down, draw a box on the map. Releasing the mouse button will cause the map to be reloaded, zoomed out to the extent of the box that was drawn.

File name	Functional tool	Referenced in	Details
transparent.gif		browse.jsp browse_results.jsp browsing.jsp details_output_include.jsp details_toolbar.jsp error_handler.jsp find_place.jsp find_place_header.jsp help_search.jsp map_title.jsp map_view.jsp searching.jsp search_form.jsp search_map.jsp search_results.jsp tabs.jsp toolbar.jsp	This image is used as a placeholder and a spacer between interface components. It is invisible and can be set to any size.
viewcoverage_off.gif and **viewcoverage_press.gif**		details_output_include.jsp	View Coverage Area rollover pair button is displayed in the details page. When clicked, a popup window opens, displaying a map showing the coverage area of the dataset.
viewDetails_off.gif and **viewDetails_press.gif**		data_output_include.jsp	View Details rollover pair button is displayed on the results page. When clicked, a detailed view of the dataset's metadata is shown.
viewmap_off.gif and **viewmap_press.gif**		data_output_include.jsp details_output_include.jsp	View Map rollover pair button is displayed for the results or details pages. When clicked, a map of the dataset's metadata is shown.

Metadata Explorer UI organization

Familiarity with HTML frames is important in understanding the relationship between the files that make up Metadata Explorer. Each frame displays a JSP page that works in coordination with the pages in the other frames.

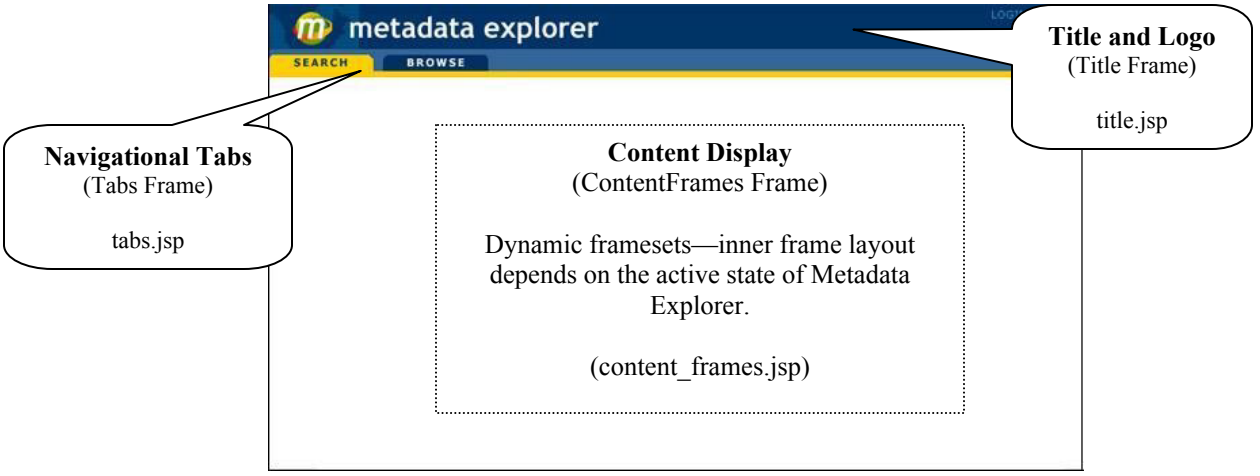


Figure 1: Identification of the initial frames

The diagram below shows the initial frame layout:

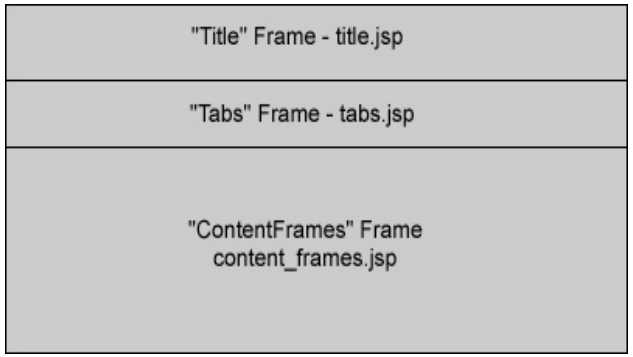


Figure 2: Initial frame layout

The **title frame** runs across the top of the browser. It holds title.jsp, which displays the logo and header image and provides the help and login/logout links if necessary.

The **tabs frame** runs across the width of the browser just beneath the title frame. It holds tabs.jsp, which displays the navigational tabs. The tabs are toggled visible/invisible and active/inactive, depending on the current state of Metadata Explorer.

The **content frame** fills up the remaining part of the screen. The inner framesets change dynamically depending on the state of Metadata Explorer. Switching between states always involves making a call back to content_frames.jsp so a new frameset can be built suited to the content that will be displayed.

Functions and dynamic changes in the content frame

The Metadata Explorer user interface changes dynamically depending on the function being performed. Most of these changes are contained within the inner framesets of the content frame. As a user moves between a standard search, gazetteer search, browse mode, or to a details or map view, the contents frame is dynamically regenerated to suit the content that needs to be displayed.

The different framesets for the various functional modes of Metadata Explorer are described in the following sections.

Search mode

The initial Metadata Explorer interface that the user interacts with is standard search mode, as shown here.

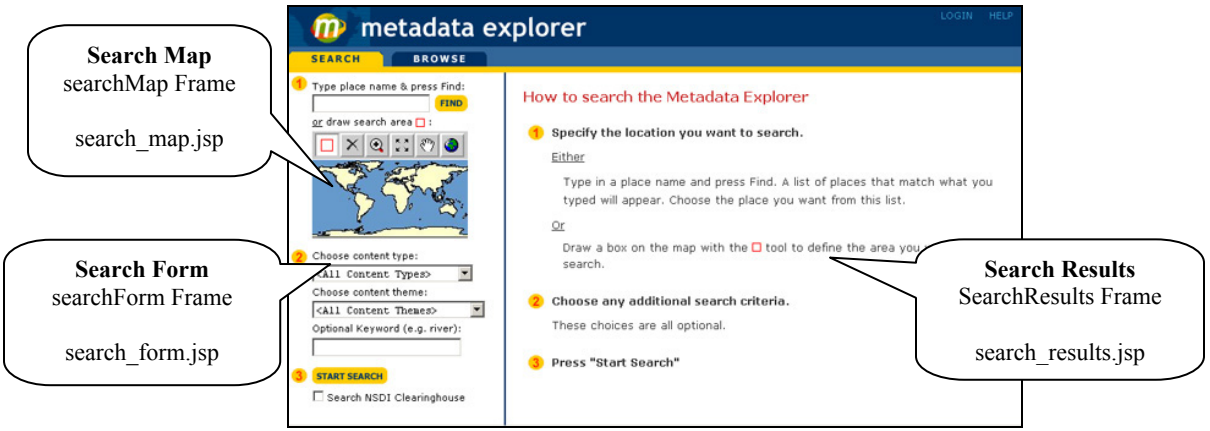


Figure 3: Identification of search frames

The diagram at right shows the frame layout when in standard search mode. The content frame is broken into two columns. The left-hand frame is further split into two rows. The top row shows the search map and, if gazetteer searching has been enabled, a text box for typing in a gazetteer search place name. The bottom row displays the search form.

Initially, the right-hand frame is only one frame, holding the search instructions file, help_search.jsp (Fig. 4). Once results are received from the server, the search results frame splits into two rows. The top row holds the search result toolbar, and the bottom row displays the search results (Fig. 5).

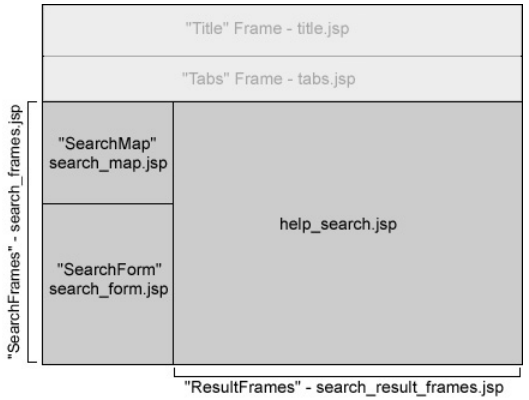


Figure 4: Initial frame layout for search mode

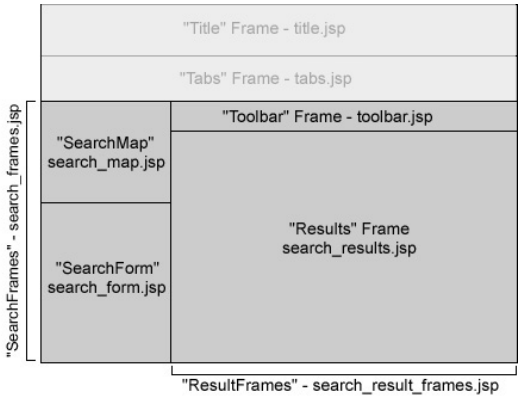


Figure 5: Frame layout for search mode when results are displayed

Gazetteer Search

If the gazetteer search functionality is enabled in Metadata Explorer, there may be more than one possible match for the place name entered by the user. The interface is modified to accommodate the additional step—choosing the search place from the possible matches—as shown here.

Figure 6: Identification of gazetteer search frames

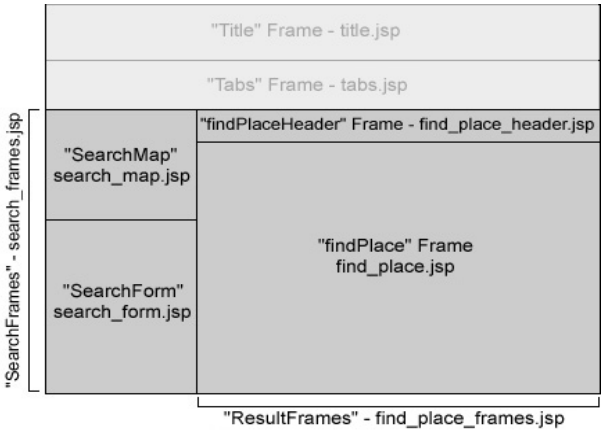


Figure 7: Frame layout for gazetteer search

The diagram at left shows the frame layout when in the gazetteer state. The content frame for the gazetteer search is similar to that of a standard search. The frame is broken into two columns. The left-hand frame is further split into two rows. The top row shows the search map and a text box for typing in a gazetteer search place name. The bottom row displays the search form. The right-hand frame is also split into two rows. The top row displays information and instructions regarding the search, for example, how many place names were found. The bottom row holds the list of hyperlinked place names matching the search.

Browsing

In browse mode, Metadata Explorer appears as at right:

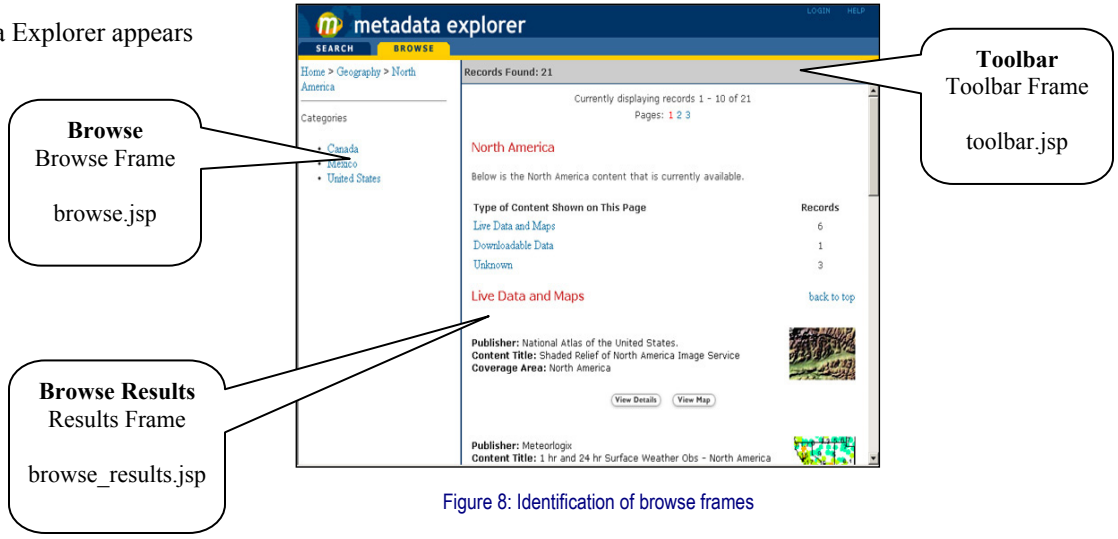


Figure 8: Identification of browse frames

The diagram at right shows the frame layout when in the browse state. The content frame is broken into two columns. The left-hand column displays the path of the browse tree, starting from the root. The right-hand column is split into two rows. The top row displays the search result toolbar. The bottom row, in the remaining space, displays the browse results.

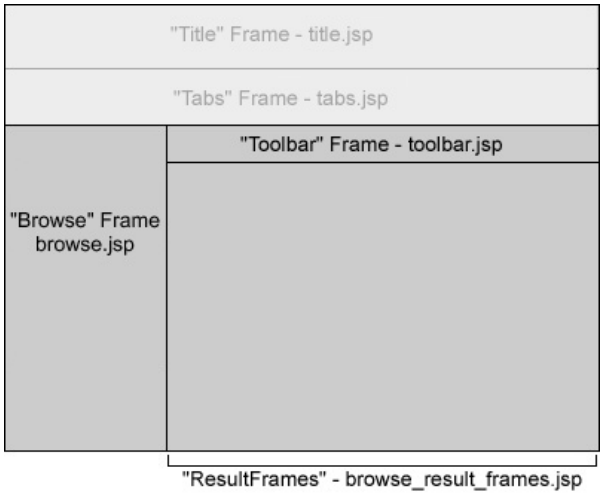


Figure 9: Frame layout for browse mode

Details View

The details view is enabled when a user clicks the Details tab.

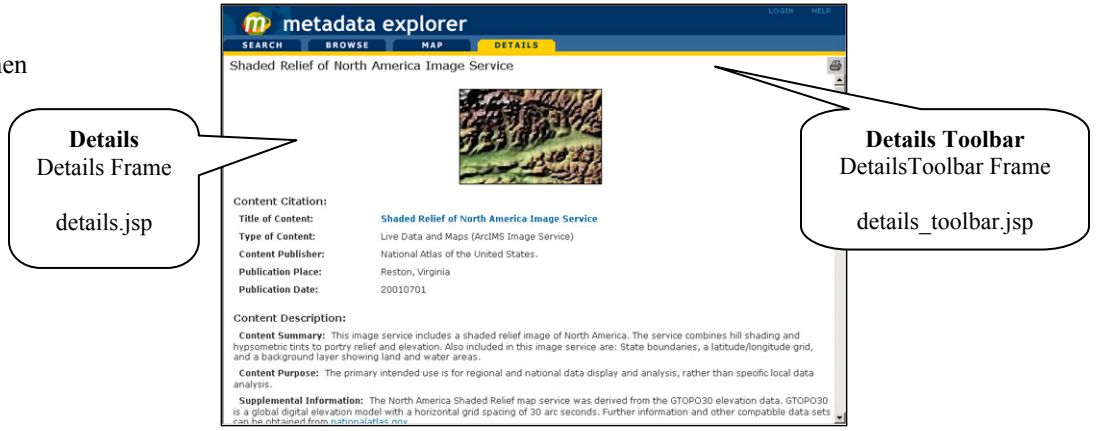


Figure 10: Identification of details frames

The diagram at right shows the frame layout when in the details state. The content frame is broken into two rows. The top row displays the details toolbar (details_toolbar.jsp). The bottom row, in the remaining space, displays the details of the dataset.

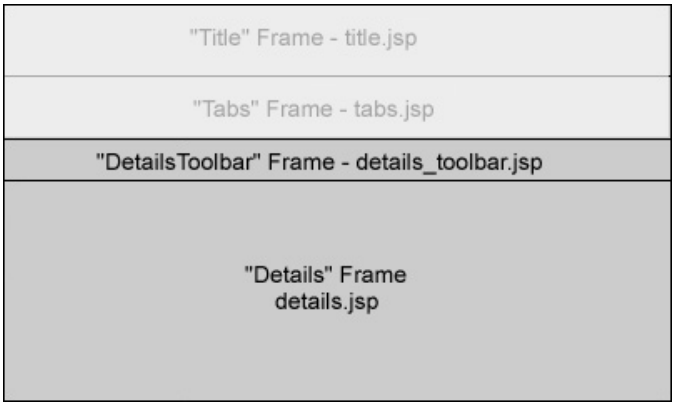


Figure 11: Frame layout for details view

Map View

As illustrated at right, clicking the Map tab changes the interface to display the map of the selected dataset.

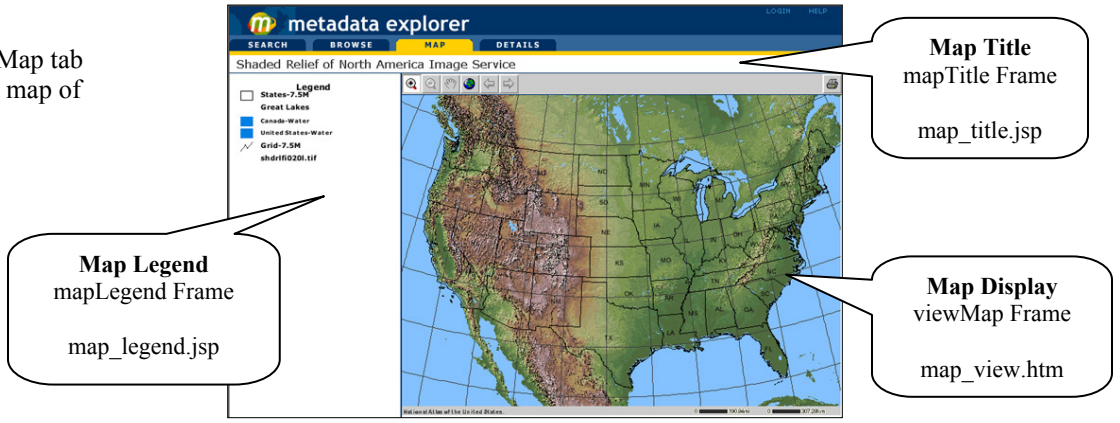


Figure 12: Identification of map frames

The diagram below shows the frame layout when in the map view state:

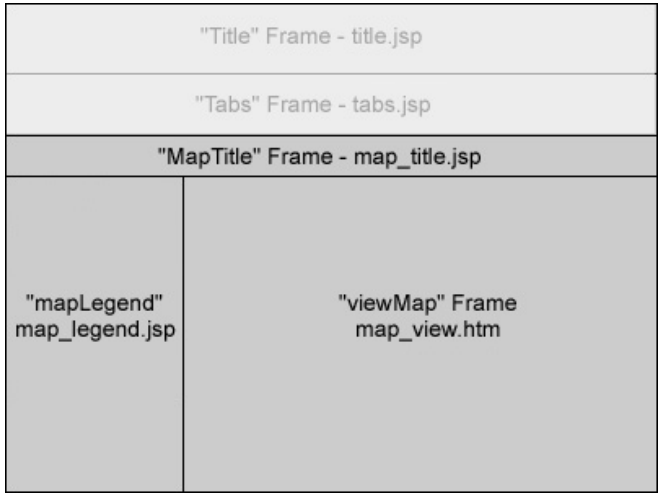


Figure 13: Frame layout for map view

The content frame is broken into two rows, with the bottom row being further divided into two columns. The top row displays the map title. The bottom row takes up the remaining screen space. The left column holds the map legend, and the right column holds the actual map image and the map toolbar above it.

Customizing Metadata Explorer 2

IN THIS CHAPTER

- **Installing the Advanced Metadata Explorer sample**
- **About the sample**
- **Step-by-step guide for basic customizations**

The Metadata Explorer application was created with the idea that it will be customized to suit the specific needs of your organization. As such, a novice user can easily change the appearance of the interface by substituting graphic files, modifying style sheets, or editing property files; a user familiar with JavaScript and HTML can quickly add or subtract functionality; and an advanced user can create a unique application from scratch using the Java Connector Object Model or Tag Library. This chapter focuses on customizations that can be made to the existing Metadata Explorer code through modifications to graphics, style sheets, property files, HTML, and JavaScript. For detailed information on creating an application using the Object Model or Tag Library see *Customizing ArcIMS—Java Connector*.

The Advanced Metadata Explorer sample introduces you to the variety of ways that Metadata Explorer can be tailored to meet the requirements of your site. The first two sections of this chapter instruct you on installing the sample and highlight the steps taken to achieve its different look and feel and to extend the search and browse functions. The final section guides you step-by-step through some of the more popular customization possibilities. Use this chapter as your road map to the Metadata Explorer you need.

Installing the Advanced Metadata Explorer sample

This sample, like the original Metadata Explorer, requires that a Metadata Service has been created and is running correctly. Additionally, this document discusses the gazetteer functionality. If you are interested in this optional feature, the gazetteer data must be installed and loaded into ArcSDE™, and you must have created a service. *Creating and Using Metadata Services* gives detailed instructions for performing all of these tasks.

Installation of the sample files

- **Windows® and Solaris®:**
  Install the sample by selecting Samples and Advanced Metadata Explorer during the ArcIMS installation program.

- **AIX®, HP-UX®, Linux®, and SGI™:**
  Install the sample by choosing it during the installation script.

  Once installed, the sample can be found in <ArcIMS Installation Directory>/ArcIMS/Samples/AdvancedMetadataExplorer.

Postinstallation setup

Some additional postinstallation steps are necessary to get the sample up and running; the following steps are also provided in the readme.txt file in the AdvancedMetadataExplorer directory.

1. Create a virtual directory named AdvancedMetadataExplorer.

   The specific steps required to create a virtual directory are dependent on the Web server and servlet engine being used. If you are unsure how to do this, *ArcIMS Installation Guide* provides instructions for creating virtual directories for the standard Metadata Explorer application. Use these steps as a guide but substitute the virtual directory/alias name (AdvancedMetadataExplorer) and file directory locations (<ArcIMS Installation Directory>/ArcIMS/Samples/AdvancedMetadataExplorer) as needed.

2. Create an Image Service named CanadaMap that references the configuration file, canada.axl, located in <ArcIMS Install Directory>/Samples/TutorialData/AXL. For instructions on creating a service in ArcIMS, see *Using ArcIMS*.

   Note: If you installed the samples into a location other than the default installation directory (C:\Program Files\ArcGIS\ArcIMS\Samples) or are a UNIX user, you need to modify the <SHAPEWORKSPACE> element of the configuration file to accurately reflect the location of the sample data before creating the Image Service.

   To do this, open canada.axl in a text editor, find <SHAPEWORKSPACE>, and modify the data path as needed.

3. In a text editor, open the aimsmeta.properties file located in the AdvancedMetadataExplorer directory that you created or mapped to in Step 1. It is located in \AdvancedMetadataExplorer\WEB-INF\classes.

   a. Change the meta_host_name property to match the machine name where your Metadata Service is running. If necessary, change the meta_service_name and meta_port_number.

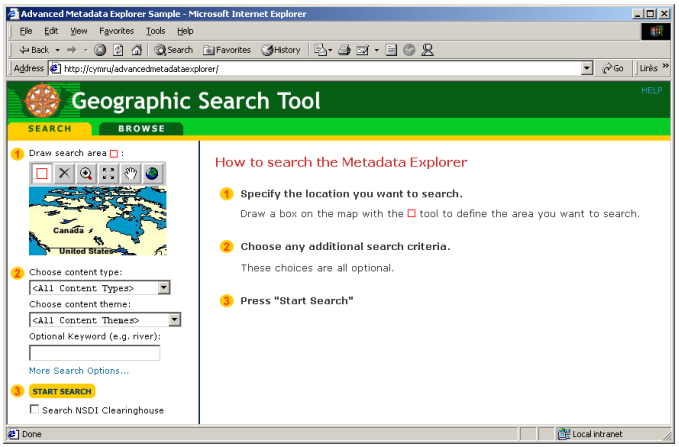
   b. Change the search_service_name to CanadaMap. Make sure that the name and spelling exactly matches the name of the Image Service you created in Step 2. The name is case sensitive.

   c. Change the search_host_name to match the machine name where you created the CanadaMap service. If necessary, change the search_port_number property.

   d. If you have a gazetteer service available and the search_gazetteer property is set to true, change the gazetteer_host_name property to match the machine name on which your gazetteer service is running. If necessary, change the gazetteer_port_number property.

4. Restart your Web server, servlet engine, and ArcIMS.

5. View the sample in a browser window. http://localhost/AdvancedMetadataExplorer



About the sample

This section describes the advanced features that were added to the Metadata Explorer and details the changes that were made to the related files.

A custom name

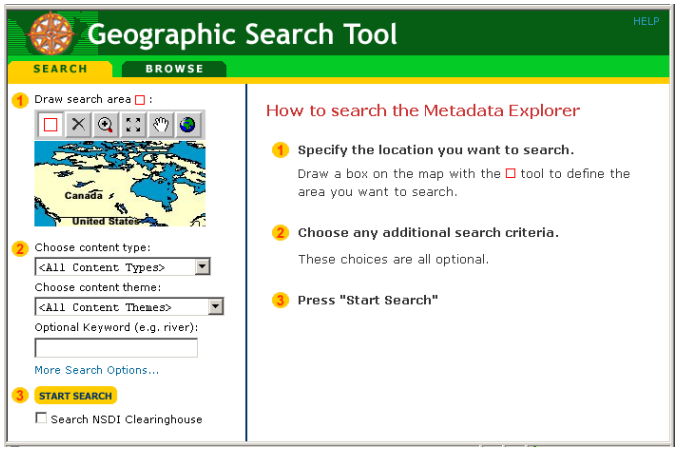
The name of the Metadata Explorer application was changed to Geographic Search Tool. To make this modification, the graphic files me_title.gif and meheader_lg.gif were changed.

Different colored graphics

Metadata Explorer took on a different look through a new color scheme. To make this change, these graphic files were altered.

- Files defining the navigational tabs that appear near the top of the interface:
  browse_tab_off.gif
  browse_tab_select.gif
  details_tab_off.gif
  details_tab_select.gif
  map_tab_off.gif
  map_tab_select.gif
  search_tab_off.gif
  search_tab_select.gif

- The button appearing on popup window pages. Clicking it closes the popup window:
  close_button.gif



Search map changed

The world map used to specify a geographic extent in the original Metadata Explorer's search mode was changed to a base map of Canada. To make this change the following steps were taken:

1. An Image Service named CanadaMap was created in ArcIMS Administrator. The configuration file (canada.axl) for this service is available in the <ArcIMS Installation Directory>/ArcIMS/Samples/TutorialData/AXL.

2. This new Image Service was added into the existing Access Control List (ACL) file (aimsacl.xml) as a freely available service—no username and password required.

3. The search_service_name property in aimsmeta.properties was altered to specify that the Image Service to be used in the search window was CanadaMap, instead of the default SearchMap.

Gazetteer enabled

By default, gazetteer searching is not enabled in the original Metadata Explorer search interface and gazetteer data is not installed and loaded in ArcSDE. However, if you have enabled this optional feature, this sample illustrates how it is set up.

To enable gazetteer searching the following steps need to be taken:

1. The ArcIMS installation process was run with the option to install gazetteer data selected.

2. The gazetteer data was imported into ArcSDE using available import scripts.

3. An additional Metadata Service was created for the gazetteer.

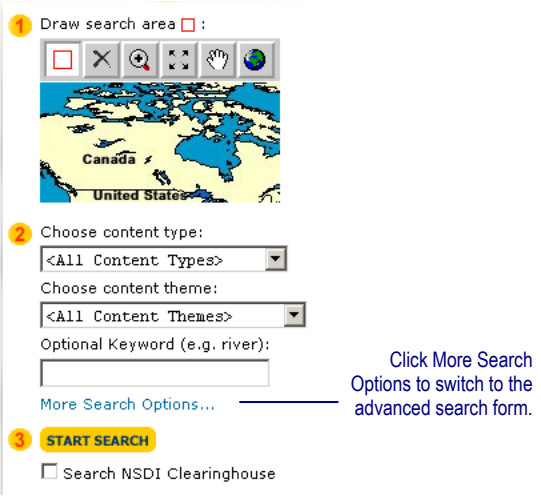
4. This service was added as a freely available service to the ACL file (aimsacl.xml).

5. The search_gazetteer property in the aimsmeta.properties file was set to true.

Detailed instructions for each of the steps outlined in the procedure above are available in *Creating and Using Metadata Services*.

Advanced search form

The search interface was altered to include both the basic search and a link to an advanced option—More Search Options.



This customization involved two steps—adding links and functions between the original search and the new advanced options search and creating the advanced search form itself.

Add More Search Options and Less Search Options links

The first step was to modify search_form.jsp to include a link to a different form with additional search options, search_form_advanced.jsp. In addition, a link pointing back to search_form.jsp was needed in search_form_advanced.jsp to give users a link offering less search options.

In order to create a better experience for the user, any criteria defined by the user that is common to both search forms is

retained when switching back and forth between them. To enable this feature, switchToSimple() and switchToAdvanced() JavaScript functions were created in search_form.jsp and search_form_advanced.jsp respectively; these functions store the values of the defined search criteria in the URL parameter list prior to switching between the search forms.

In addition, the function saveSearchArgs() in both search_form.jsp and search_form_advanced.jsp was modified to include a new parameter in the search URL arguments. This parameter indicates whether the search was initiated from the simple or advanced form and ensures that the correct interface is restored when a user returns to search mode from a details or map view.

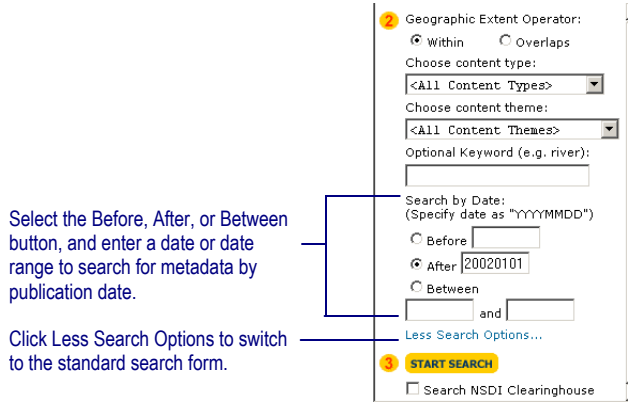
Additional modifications were made to the saveSearchArgs() function in search_form_advanced.jsp so that it could store as parameters the advanced search criteria defined by the user. Finally, changes were made to prevent the inclusion of advanced criteria if a National Spatial Data Infrastructure (NSDI) Clearinghouse search is initiated.

Features included in the advanced search form

Clicking More Search Options from the Search interface of the sample reveals the two advanced search criteria that are available—search for data that falls entirely within or overlaps the boundaries of the geographic extent defined for the search and focus the search by publish date.

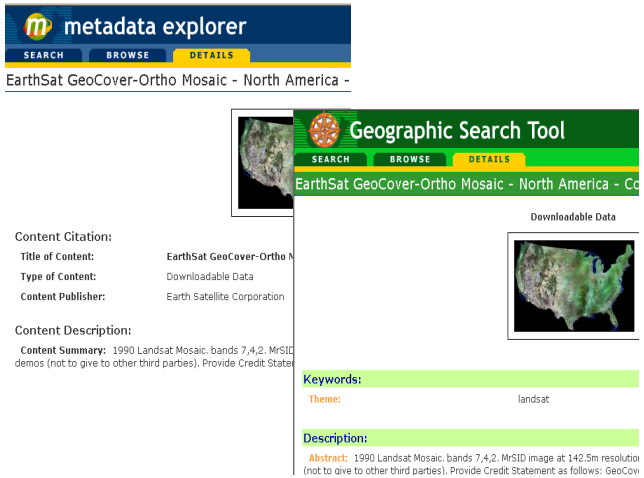
To give users the ability to search for data that falls within or overlaps the defined geographic area, HTML radio buttons were added to the form in search_form_advanced.jsp. Search_results.jsp was modified to read the extentOperator parameter, the value of the selected radio button from search_form_advanced.jsp, into the spatialOperator attribute of the <aims:searchEnvelope> tag. If the extentOperator parameter was not defined, the default value for the operator is 'overlaps'.

The ability to specify a date search by selecting a Before, After, or Between HTML radio button and entering a date in the text boxes was added to the form in search_form_advanced.jsp. Search_results.jsp was modified to read in the values of the advanced search form's date elements. In addition, a conditional <aims:valueQuery> tag, based on the kind of date search to be performed, was added to the body of the <aims:search> tag. This enables the date criteria to be included in the search request. The date search is performed on the Federal Geographic Data Committee (FGDC) publication date tag, metadata/idinfo/citation/citeinfo/pubdate.



Details display enhanced

Reorganizing the information and adding color to the display enhanced the details page.



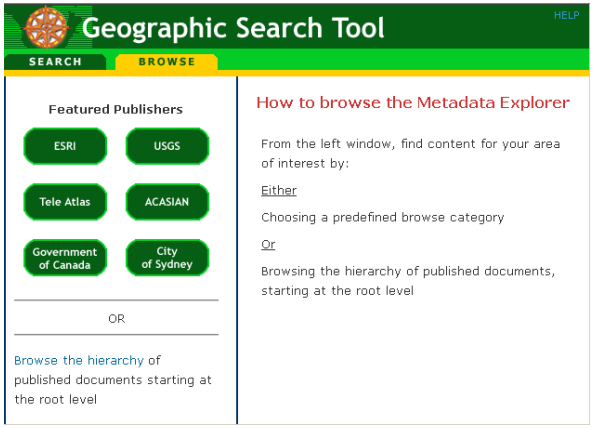
Color was added to the title bar and section headings, the section headings themselves were renamed, and the organizational appearance of the details view was modified from the original Metadata Explorer.

To make this change, the following files were altered:

- A background color was added to the title row by modifying details_toolbar.jsp.
- Different colors were added to the headings by modifying the metadata.css style sheet file. All font definitions for the details page are named starting with the word "details".
- Section headers were renamed by making changes to res.properties.
- The HTML code in details_output_include.jsp was altered to give the details page a different look and organization. For information on changing the actual content displayed see 'Changing the look of the details page' later in this chapter.

Browsing by predefined categories

Additional functionality was added giving users the ability to browse through available published documents via predefined categories.



Buttons were added that enable browsing for metadata published by select organizations. The existing hierarchical browse capability is also available.

The following files were altered or created to enable this functionality:

- A link was added to the top of the hierarchy folder listing to give users the ability to switch to Browse by Predefined Categories mode by modifying browse.jsp.

- The right-hand pane displays either an introduction to the browsing methods available or the results from a browse request. In order to build the dynamic frameset and content required by user interaction with the interface, browse_result_frames.jsp was modified.

- browsing.jsp was modified to determine via a new predefined parameter in the URL whether a category or hierarchal browse was selected. If the request is for the traditional hierarchal browse, the page forwards its location to browse_results.jsp. If the request is for a specified category browse, the page forwards its location to browse_results_predefined.jsp.

- A new file, browse_results_predefined.jsp was created. This file sends category browse requests—search requests with predetermined search criteria—to the server. Upon receiving a response, browse_results_predefined.jsp displays the list of datasets meeting the selected browse criteria.

- A new file, browseChoice.jsp was created. It displays the hyperlinked category buttons and provides a link for initiating a hierarchal browse. This file is displayed in the left-hand pane of the interface frameset when a new browse is initiated (i.e., user switches from search mode to browse mode) and when the user is browsing via the predefined categories.

- content_frames.jsp was modified to either load browse.jsp or browseChoice.jsp in the right-hand pane of the browse frameset depending on the value of the goTo parameter.

- A new file, help_browse.jsp was created. This file displays brief user instructions detailing how the Metadata Server can be browsed. When a new browse is initiated, help_browse.jsp is displayed in the right-hand pane of the interface frameset.

- print.jsp was modified to include the retrieval of results for a predefined browse request.

- Res.properties was modified to include resource bundle strings for browse_results_predefined.jsp, browseChoice.jsp, and help_browse.jsp.

- The goTo parameter in tabs.jsp was modified to allow its value to change from browse to browseNew when a user switches from search mode to browse mode.

- Six new graphic files were added to the Images directory, one for each category button displayed in browseChoice.jsp. The files are acasian.gif, cityofsydney.gif, esri.gif, governmentofcanada.gif, teleatlas.gif, and usgs.gif.

Step-by-step guide for basic customizations

This section gives step-by-step instructions for customizing Metadata Explorer in a variety of ways. Some of these options were applied to the sample described in Section 1 of this chapter.

Changing access options

Modifications in this section include:

- Changing the Metadata Service associated with Metadata Explorer application.
- Restricting access to metadata documents or ArcIMS services that are part of your metadata catalog.
- Using HTTP as the connection protocol.
- Integrating Metadata Explorer into an existing Web site or application.

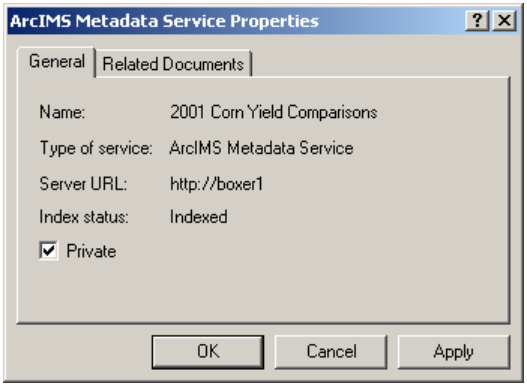
Change the Metadata Service that is searched
The Metadata Service that is searched and browsed via Metadata Explorer is specified in the aimsmeta.properties file. To change the service associated with a particular Metadata Explorer application:

1. In a text editor, open its aimsmeta.properties (/WEB_INF/classes) and locate the property meta_service_name.
2. Change this property's value to the name of the Metadata Service that you would like Metadata Explorer to be associated with.
3. The meta_host_name and meta_ port_number property values must equal the name and port number of the server hosting the Metadata Service. If necessary, edit these values.
4. Save and close the file.

Create and view private documents
Access to individual metadata documents can be restricted so that only those users who have a valid username and password combination are able to view them. Any published metadata document can be marked as private, making it invisible to unauthorized users.

To mark a document private:

1. In ArcCatalog™, connect to the Metadata Service using a valid username and password combination.
2. Right-click the published document that you want to make private.
3. From the dropdown menu, choose Properties.
4. Select the General tab in the Properties dialog box.



5. Click the Private box to activate private status.

6. Click OK to close the properties dialog box.

The document, now marked private, is visible only to those users who login to Metadata Explorer using a valid username and password. The only valid username and password combination is the one used to connect to the Metadata Service (as in Step 1 above) when initially marking the document as private.

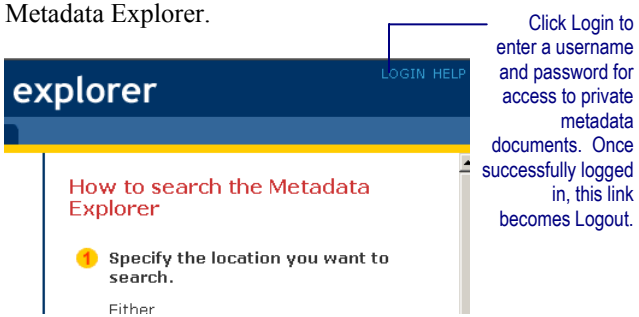
When a document is made private, it is not visible to the general user searching or browsing your Metadata Service via Metadata Explorer. In order to see private documents, a user must log in to Metadata Explorer with the correct username and password. By default, Metadata Explorer does not include a login interface. This feature is available but must be activated.

To enable Metadata Explorer's login interface:

1. In a text editor, open aimsmeta.properties (/WEB_INF/classes), locate the login property, and set its value. This property can be set to one of three values: off, optional, or required.

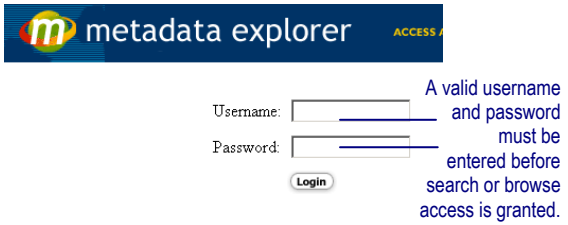
   - The default setting, off, does not give the user the ability to log in. Any documents that are marked private will not be seen in any Metadata Explorer results.

   - Setting the value to optional gives the user the choice of logging in or accessing Metadata Explorer as a guest. When the login property is set to optional, a Login link appears in the top right-hand corner of the Metadata Explorer.



     Clicking this link takes the user to a screen for inputting a username and password. When a login is successful, the Login link changes to a Logout link, and the user is able to view any documents that were marked as private in ArcCatalog using that username and password.

- Setting the value to required forces anyone who attempts to access Metadata Explorer to log in with a valid username and password. The login interface becomes the initial screen of Metadata Explorer.



  Once successfully logged in, the search and browse interfaces of Metadata Explorer are enabled and any documents that were marked private in ArcCatalog, using that username and password, will be visible.

2. Save and close the file.

Setting the login property of aimsmeta.properties to optional or required activates the login interface and links in Metadata Explorer. The next step is to enable the validation process and tell Metadata Explorer where to find the list of acceptable username and password combinations.

Metadata Explorer communicates with ArcIMS via the Java Connector. The file authenticate.properties is used to set parameters for user authentication in the Java Connector.

1. In a text editor, open authenticate.properties (WEB-INF/classes), locate the authenticate property, and set its value to true.

2. Find the aclFileName property and set its value to the file path location of your ACL file.

   The ACL file you specify could be one which is used exclusively by Metadata Explorer or one that is used to control access to other ArcIMS functionality as well. For instance, an ACL is necessary to publish documents to a Metadata Service. For a more detailed discussion, see Appendix A in *Creating and Using Metadata Services*.

   Note: Metadata Explorer also supports the use of a Java Database Connectivity (JDBC)-based ACL. For instruction, see *ArcIMS Help*.

In addition, in order for login functionality to be operative, any user accessing Metadata Explorer must have cookies enabled.

Restrict access to published ArcIMS services

ArcIMS services that have been published to a Metadata Service can be viewed in Metadata Explorer. Unlike standard metadata documents, these services cannot be made private, and, therefore, invisible, using ArcCatalog. However, while published services cannot be excluded from the results list of a search or browse request, View Map access to these services can be restricted to users who provide a valid username and password. This functionality is controlled via the ACL.

Since the ACL file lists all available services, simply remove the service name from the list of freely available services and add it as a restricted service under the username and password desired.

Following is an example of the syntax used in the ACL file to protect an ArcIMS service:

```
<USER name="redlands" password="redlands"
services="RedlandsStreets
roles="metadata_browser" active="1" />
```

In the example, when the published RedlandsStreets Image Service is returned in a result set, clicking View Map below the service listing sends the user to a page for the input of a username and password. The user must input the username/password combination of redlands/redlands in order to view the Image Service map.

Toggle between http or tcp connections

Each JSP page in Metadata Explorer that sends a request to the server first has to set up a connection with the server. Connections can be made via two different methods:

- TCP: When using a Transmission Control Protocol (TCP) connection, you must specify the host name and port number of the server you are connecting to. You can set the host name and port number of any service that you are connecting to (Metadata Service, SearchMap Image Service, or Gazetteer Service) in the aimsmeta.properties file. TCP is the recommended connection method if your server is located internally on the network and you do not have any firewalls to cross.

- HTTP: If you do not have the information necessary to make a TCP connection, you can choose to connect to the server via the hypertext transfer protocol (HTTP). An HTTP connection only requires that you specify the URL of the server you are connecting to and is the recommended method to use if you are going outside of a local network or across firewalls to connect to the server.

In a default installation of the Metadata Explorer, all JSP pages connect to the server using TCP connections. The only exception is map_view.jsp and map_print.jsp, which make an HTTP connection to the Image Service for the map they will display. In all other pages, change the connection type from TCP to HTTP via the following modifications:

1. In each JSP file that makes a connection to the server, locate the <aims:tcpConnection> tag. You can find this tag in each file using a text search with the keyword aims:tcpConnection.

   The following is a list of all the files that make a connection to the server and the line numbers on which you can find the <aims:tcpConnection> tag:

   - add_relevance.jsp: line 13
   - browse.jsp: lines 36, 45, and 51
   - browse_results.jsp: lines 37, 46, and 52
   - browse_results_predefined.jsp (a new file created for the Advanced Metadata Explorer sample): lines 37, 46, and 52
   - coverage.jsp: line 42
   - details_output_include.jsp: lines 36, 45, and 51
   - find_place.jsp: line 22
   - login.jsp: line 44
   - map_view.jsp: lines 36, 45, and 51
   - print.jsp: lines 36, 45, and 51
   - search_map.jsp: line 99
   - search_results.jsp: lines 37, 46, and 52

2. Change the <aims:tcpConnection> tag in each of the places listed above to the <aims:httpConnection> tag.

3. Remove from each tag the host and port number attributes and replace them with the URL attribute specifying the URL of the server you are connecting to.

4. If necessary, include other optional attributes that you may need to make a successful connection. A list of attributes for the <aims:httpConnection> tag is available in the Tag Library section of *Customizing ArcIMS—Java Connector*.

Integrate the Metadata Explorer into a site or application
Metadata Explorer is a stand-alone browser application. It is accessed via a URL specifying the machine name hosting it, followed by the virtual directory name containing the Metadata Explorer files. An example of a Metadata Explorer URL is:
`http://mymachine.esri.com/MetadataExplorer`

Metadata Explorer, however, can be integrated within a bigger application or Internet site. An example of this is the Geography Network (http://www.geographynetwork.com). On the Geography Network site, one can launch the Metadata Explorer into different modes from a variety of links such as directly to a map or details view or to results of a keyword-specific search.

Examples of URLs and their associated parameters that launch Metadata Explorer into specific modes or from different applications are provided in spawner.html.

The following URLs show how to launch Metadata Explorer into different modes:

- *explorer.jsp*—without parameters attached, launches Metadata Explorer into its starting mode with the search tab selected and search instructions in the right-hand pane of the different options.

- *explorer.jsp?goTo=details&docId=%7B06686DD0-10FC-4BE8-9276-513D979A85F1%7D&hasMap=false*—launches Metadata Explorer into the Details mode, showing the details of the dataset identified by the docID parameter. If the hasMap parameter is set to false, the Map tab will not be visible. If this dataset has an associated live map that is an Image or Web Map Server (WMS) Service, set the hasMap parameter to true. Note: only ArcMap Image, standard Image, and WMS Services can be displayed in the Metadata Explorer. The Java Connector cannot stream Feature Services.

- *explorer.jsp?goTo=map&docId=%7BF2FC2F29-66CD-4DE7-859A-9B99FC2218FD%7D*—launches Metadata Explorer into the Map mode, showing a map of the dataset identified by the docID parameter.

- *explorer.jsp?goTo=search&extent=-124.39,32.53,-114.12,42&keyword=river*—launches Metadata Explorer into a search mode with a specific extent and keyword.

The following URLs show how to launch Metadata Explorer from another application:

- *explorer.jsp?mode=arcmap*—Metadata Explorer launches from within ArcMap. When Metadata Explorer is in the ArcMap mode, the Add to ArcMap button will appear in a results view below datasets that have a content type of Live Data and have the Geography Network Document (GND) property defined.

- *explorer.jsp?mode=arcmap&extent=-100,-50,100,50*—Metadata Explorer launches from within ArcMap and

with a specified extent defined and drawn on the search map.

- *explorer.jsp?mode=arcexplorer&port=-1*—Metadata Explorer launches from within ArcExplorer. When Metadata Explorer is in the ArcExplorer mode, the Add to ArcExplorer button will appear in a results view below the datasets that have a content type of Live Data and have the Geography Network Document property defined.

- *explorer.jsp?mode=viewer&port=-1*—Metadata Explorer launches from within a viewer. When Metadata Explorer is in the Viewer mode, the Add to Map button will appear in a results view below the datasets that have a content type of Live Data and have the Geography Network Document property defined.

Changing the look of Metadata Explorer

Changing the look of the Metadata Explorer can involve two types of changes:

- Visual changes such as the color scheme, logo, and title

- Interface usability changes such as changing the layout or the frameset organization

Change the color scheme, logo, or title
To make changes to the color scheme, logo, or title of the Metadata Explorer, graphic files located in the Images subdirectory need to be modified. Examples of files that would need to be changed for various parts of the interface are:

- **me_title.gif** is referenced in title.jsp and is the main bar appearing on the top of the interface. It needs to be modified to make a change in the logo, title, or color scheme.

- **meheader_lg.gif** is referenced in login.jsp, map_print.jsp, and print.jsp and is the default title bar on pages that are not part of the main Metadata Explorer interface such as the popup print pages. It needs to be modified to make a change in the logo, title, or color scheme.

- **browse_tab_select.gif** and **browse_tab_off.gif, details_tab_select.gif** and **details_tab_off.gif, map_tab_select.gif** and **map_tab_off.gif, search_tab_select.gif,** and **search_tab_off.gif** are used to display the navigational tabs of the interface. Each pair of images creates one navigational tab with a rollover effect when a mouse moves over the tab. Depending on the mode that Metadata Explorer is in, the visibility of each pair of files is toggled. These files would need to be modified to make a change in the color scheme.

- **viewcoverage_press.gif** and **viewcoverage_off.gif, viewmap_press.gif** and **viewmap_off.gif, viewdetails_press.gif** and **viewdetails_off.gif** are used to display the buttons that appear underneath each dataset record in a result set. Each pair of images creates one button with a rollover effect when the mouse moves over the button. These files would need to be modified to make a change in the color scheme.

- **tool*.gif—**all images whose names start with tool are used as map tools, whether it is for the search map used to define geographic search extents or the map view for a dataset. These files would need to be modified to change the look of the tools.

- **find_off.gif, find_press.gif, search_off.gif, search_press.gif, search_1.gif, search_2.gif,** and **search_3.gif** are buttons or images that appear when in search mode. They would need to be modified to make a change in the color scheme.

Change the interface layout
The interface of Metadata Explorer is built as a series of frames and subframes, as described in Chapter 1. Changing the interface layout involves changing the various files that define the different framesets:

- explorer.jsp—change the frameset in this file to alter the parent layout of the entire interface, which includes the title bar, navigational tabs, and the content frames
- content_frames.jsp—change the frameset in this file to alter the layout of the content frame and its subframes
- search_frames.jsp, search_result_frames.jsp—change the framesets in these files to alter the layout of the interface when in a search mode
- find_place_frames.jsp—change the frameset in this file to alter the layout of the gazetteer search results
- browse_result_frames.jsp—change the frameset in this file to alter the layout of browse results

Changing the look of the details page

All modifications to the details page can be done by modifying details_output_include.jsp.

Include different metadata elements
Metadata Explorer, by default, makes select metadata elements from the record's entire metadata information viewable when the user clicks View Details. The elements displayed can be modified.

1. Decide which metadata elements you want to display and determine their exact XML path. You can open the metadata file to find the XML path and ArcCatalog offers an XML style sheet to view the metadata elements.

   For this example, the metadata element Native Form will be used. Its XML path is metadata/idinfo/native.

The next three steps involve edits to the details_output_include.jsp file.

2. Open the details_output_include.jsp file. Append the new metadata element to the list of elements already used. This list of elements is stored in a string, approximately line 135. To find this list using a text search, search aims:loadElements; the list is above this function.

   In this example, the XML path to Native Form will be added after the Use Constraints element. The Native Form will be added to the FGDC string elements.

   ```
   + "metadata/idinfo/useconst,"
   + "metadata/idinfo/native,";
   %>
   ```

   All the values of the elements specified in the string (from #2) are loaded into memory. Whatever new element you specify in the string will be loaded as well. This is done via the <aims:loadElements> JSP tag that you can see at approximately line 199.

3. Starting on line 203 through line 309, you will notice code that pulls out the values of these memory-loaded elements and stores them as variables on the page. Add

code, using the <aims:getElement> tag, to pull out these values of your elements, storing them as variables on the page. The syntax is as follows:

```
<aims:getElement
id="assign_a_variable_name_here"
loadedElements="<%=theElements %>"
value="the_XML_path_to_your_element" />
```

The two things you need to change are the id and the value attributes of this tag. The id is whatever name you want to give to your variable, and the value is the XML path of your element, the string you specified in Step 2. The syntax for this example is:

```
<%-- Get the dataset's native form --%>
<aims:getElement id="theNativeForm"
loadedElement="<%=theElement%>"
value="metadata/idinfo/native" />
```

4. Now that you have a variable name storing the value of your element, you can use it anywhere on the page. To print the value use the syntax `<%=variableName%>` anywhere in the HTML code of details_output_include.jsp. In this example, this line could be added:

```
The Native Form is: <%= theNativeForm %>
```

Change the visual display

The look of the details display page is defined by HTML code within the page. Changing the layout of the details display, changing the color scheme, or the order in which metadata elements are displayed can all be done by modifying the HTML code. Font styles used within the page are defined in metadata.css.

Change the display of related documents

The names of a dataset's related documents are shown in a hyperlinked list at the end of a details display for the dataset. Clicking on one of these hyperlinked names takes you to a details display for the related document. To modify the display of the related records list:

1. You will find the code for displaying related results on about line 916 of details_output_include.jsp. To find this section of code using a text search, search "related documents"—the code is just below this string.

   The related documents are first retrieved via a browse request, <aims:browse>, specifying to return the siblings of the current dataset.

2. On line 929 (line 933 in the advanced sample), <aims:iterateMetadata> iterates through the result set of related documents, printing out the hyperlinked name of the datasets.

   To customize the list of related documents, retrieve any dataset property you wish to display using the <aims:datasetProperty> tag. Dataset properties are information about a dataset stored by the Metadata Server. The list of valid properties is referenced in the Tag Library section of *Customizing ArcIMS—Java Connector*. The syntax for retrieving a dataset property is as follows:

   ```
   <aims:datasetProperty
   id="assign_a_variable_name_for_property"
   value="one_of_the_valid_properties">
   ```

   If you wish to display information about a related document that you cannot retrieve using the <aims:datasetProperty> tag, you can use the combination

of the <aims:loadElements> and <aims:getElement> tags to retrieve any information you want from the dataset's metadata. Details on how to retrieve information using these two tags is available in the section 'Include different metadata elements' earlier in this chapter.

3. Once you have retrieved all the properties and metadata elements you wish to display for the related dataset, write them out to the page using HTML code and the syntax

```
<%= variableName %>
```

Change the buffer size of the coverage area

1. In display.jsp locate the hightlight() function on line 46. To find the function using a text search, search function hightlight().

2. The fifth line of the function calls the addPercent function with a specified percentage passed in as a parameter.

3. Change this number to your desired percentage.

Changing the results display

Change the order of returned results in a search or browse
By default, results are returned in the following order:

- Results from a search or a predefined category browse are ordered in ascending order of their extent area. When browsing through the hierarchy, results are ordered by their relevance factor. After this initial ordering, the results are returned to Metadata Explorer in batches of a size specified in the aimsmeta.properties file.
- Within a batch, results are ordered by the content type for both search and browse results.

To alter the sort order of search or browse results:

1. In search_results.jsp or browse_results_predefined.jsp, locate the line <aims:search>. In search_results.jsp, approximately line 298 or line 333 in the advanced sample; in browse_results_predefined.jsp, line 107 or a new file for the advanced sample. To find this code via a text search, search <aims:search>.

2. In browse_results.jsp, locate the line <aims:browse>, approximately line 94. To find this code via a text search, search <aims:browse>.

3. Both the <aims:search> and <aims:browse> tags have common attributes called sort and sort2. The sort attribute defines how the results are going to be sorted in their initial retrieval, before they are broken up into batches of a given size. The sort2 attribute defines the sort order within a batch.

4. Change the value of the sort and sort2 attributes to the sort order desired. Allowable values are:

    - *Name*—Orders the results alphabetically by the dataset name
    - *Relevance*—Orders the results by the relevance factor, in descending order (from highest to lowest)
    - *Contenttype*—Orders the results by content type
    - *Localarea*—Orders the results by area in ascending order
    - *Globalarea*—Orders the results by area in descending order

5. When printing results, print.jsp resends the search or browse request to the server and retrieves the results anew. Any changes that you make in Steps 3 and 4 must be reflected in print.jsp.

   - Changes to the sort attributes for search results need to be made to the <aims:search> tag, approximately line 307 (line 342 in the advanced sample). To find this using a text search, search <aims:search id="searchResults">.

   - Changes to the sort attributes for browse results need to be made to the <aims:browse> tag, approximately line 375 (line 421 in the advanced sample). To find this using a text search, search <aims:browse id="browseResults">.

   - Changes to the sort attributes for predefined browse results need to be made to the <aims:search> tag, approximately line 497 in the advanced sample. To find this using a text search, search <aims:search id="predefinedBrowseResults">.

Change the look of the results display

All modifications to the results display can be done by modifying data_output_include.jsp.

The code inside data_output_include.jsp is contained within two loops. The first loop, <aims:iterateContentType>, prints out the summary of results. It iterates through the datasets in a result set and checks each dataset's content type. The loop returns a count of the results in each content type. The second loop, <aims:iterateMetadata>, iterates through the datasets in the result set and prints out certain details about each one.

To change the display of the result summary:

1. Locate the tag <aims:iterateContentType> found on line 39. To find this tag using a text search, search aims:iterateContentType. This is the beginning of the loop that prints the result summary.

2. The first thing the loop checks for is whether the current result set has more than one content type. If it does not, the loop exits. If you want a summary to print even if there is only one content type, remove the if statement:

   ```
   <% if (moreThanOne.equals("true")) { %>
   ```

   on line 42, and its closing bracket `<% } %>` on line 73.

3. Next the loop constructs the table and prints the table headers. The strings that are printed are retrieved from the res.properties file. To modify the appearance of the headers, for example, to add a background color to the table headings, modify the HTML code on lines 54 to 61.

4. Finally, lines 68 to 71 print out the content type names and their count.

The content types used within Metadata Explorer are hard-coded content types used in Metadata Explorer, the Java Connector, ArcCatalog, and the Metadata Server. Modifying the list of content types is not recommended because so many different components are affected by the change. For more details, refer to 'Alter the content type dropdown list' or 'Alter the content theme dropdown list' later in this chapter.

To change the display of the results:

1. Locate the tag <aims:iterateMetadata> found on line 92. To find this tag using a text search, search aims:iterateMetadata. This is the beginning of the loop that iterates through all the datasets in a result set, printing information about each one.

2. The information that is printed out for each dataset is a combination of information retrieved via the <aims:datasetProperty> tag and the <aims:loadElements>/<aims:getElement> tag duo. Dataset properties are information about a dataset stored by the Metadata Server. If the information you want to print for a dataset is not available as a dataset property, you can retrieve it directly from the dataset's metadata using the loadElements and getElement tags. Retrieved elements are stored as variables on the page.

   When using the <aims:datasetProperty> tag, the syntax is:

   ```
   <aims:datasetProperty
   id="the_variable_name"
   value="dataset_property_to_retrieve" />
   ```

   When using the combination of the <aims:loadElements> and <aims:getElement> tag, the syntax is:

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

   <aims:getElement
   id="the_variable_name"
   loadedElements="<%= theElements %>"
   value="metadata_element_to_retrieve" />
   ```

   Details on how to retrieve information using these two tags are available in the section 'Include different metadata elements' earlier in this chapter.

3. Since the results in a batch are sorted by content type, all results of one content type will be written out sequentially before results belonging to the next content type are printed. Lines 284 through 299 determine if the loop needs to write out a content type header to indicate that the next result is of a different content type than the previous one.

4. The code for writing out the details of each dataset in the result set is located between lines 301 and 345. To alter the display of results, modify the HTML code between these lines to write out any of the properties or metadata elements that you retrieved for a dataset in Step 2. Use HTML code and the syntax

   ```
   <%= the_variable_name %>.
   ```

Create a link to the data described by its metadata

In Metadata Explorer, a user can access the data described by its metadata in a variety of ways:

- If Metadata Explorer was launched from ArcMap, ArcExplorer, or a map viewer, datasets that are ArcMap Image, standard Image, and WMS Services can be added as a layer to the application that launched them if a Geography Network Document is defined for the dataset. During publishing, the Metadata Server creates a GND file for the dataset, defining the Image or WMS Service layers.

- The FGDC element *metadata/idinfo/citation/citeinfo/onlink* or the ISO element *distInfo/distributor/distorTran/onLineSrc/linkage* defines an Online Linkage. The value of this element can be specified by the metadata author such as a URL to the company that distributes the data, a URL to a ZIP file containing the data, or even a URL to the actual data itself.

  If metadata is created using ArcCatalog, the onlink element will be automatically set to a UNC path to the data. If the data is contained within a database, the onlink attribute will specify database connection parameters.

During publishing time, the Metadata Server stores the onlink element value as information for a dataset. This value can then be stored as a variable on a page using the <aims:datasetProperty> tag, with the syntax:

```
<aims:datasetProperty id="theOnlink"
value="ONLINK" />
```

Once the value has been stored as a variable on the page, you can use the variable in any way you need to create a link to the data from the Metadata Explorer. The syntax for retrieving the variable value is: `<%= theOnlink %>`

Changing search interface options

Add extra search criteria options

The Advanced Metadata Explorer sample described earlier in this chapter has a number of additional search criteria options defined. These are:

- The ability to define the geographic search operator to include data that overlaps or is within the selected area
- The ability to specify a date search

The search form can be further enhanced to include the ability to search by virtually any criteria. The code behind the search criteria form is located in search_form.jsp. Any modifications to the search form can be done by altering this file.

In the Advanced Metadata Explorer sample, however, search_form.jsp was left mostly untouched, and a new file, search_form_advanced.jsp, was created. Details of the changes are described in 'Advanced search form' earlier in this chapter. By separating the simple form from the advanced one, the user does not need to see all the criteria and go through a long form if only wanting to define a simple search.

To add additional search options to the search form:

1. The code for the search form begins on line 78 (line 99 in the advanced sample) of search_form.jsp. To find this using a text search, search <form—the opening of the HTML form tag.

2. The remaining code in search_form.jsp creates the search form. Using HTML code, add form elements to this search form to include your desired criteria.

   When the form is submitted all form element values are passed on as URL parameters to the receiving page.

3. Search_results.jsp is the final page that receives the form values and the page that constructs the request sent to the server. The process for constructing the request is as follows:

   a. Lines 91 through 115 (lines 95 through 136 in the advanced sample) read in all the parameters from the URL and store the parameter values as variables on

the page.

Make additions to these lines to include any custom search parameters resulting from search form modifications in Step 2.

b. Lines 119 through 289 (lines 140 through 324 in the advanced sample) determine which criteria have been defined and hence should be included in a search request.

Make additions to these lines to determine if any of the custom criteria from Step 2 needs to be included in the search request.

c. Lines 298 through 323 (lines 333 through 370 in the advanced sample) construct the search request.

Make additions to these lines to include any of the custom criteria from Step 2 in the search request.

d. If the search request did not yield any results, lines 326 through 346 (lines 373 through 413 in the advanced sample) print to the screen the criteria that were defined in order to help the user redefine the query with different criteria.

Make additions to these lines to include a printout of any of the custom criteria from Step 2.

4. When printing results, print.jsp resends the search request to the server and retrieves the results anew. Any changes made in Step 3 must be reflected in print.jsp.

Alter the content type dropdown list

The content types used within Metadata Explorer are hard-coded content types used in Metadata Explorer, the Java Connector, ArcCatalog, and the Metadata Server. Modifying the list of content types is not recommended because so many different components are affected by the change.

Changing the content type involves:

1. Changing Metadata Explorer. The files that reference the list of content types are:

   - *search_form.jsp*—lines 103 through 119 (lines 124 through 139 in the advanced sample) print out the content types in a dropdown list.

   - *search_form_advanced.jsp*—lines 174 through 189 (a newly created file for the advanced sample) print out the content types in a dropdown list.

   - *data_output_include.jsp*—lines 258 through 282 translate the numerical content type code into a string.

   - *details_output_include.jsp*—lines 100 through 132 translate the numerical content type code into a string.

   - *print.jsp*—lines 138 through 208 (lines 173 through 243 in the advanced sample) translate the numerical content type code into a string.

   - *search_results.jsp*—lines 133 through 203 (lines 168 through 238 in the advanced sample) translate the numerical content type code into a string.

- *res.properties*—lines 164 through 173 (lines 194 through 203 in the advanced sample) provide the mapping from the numerical content type codes to strings.

2. Changing the IterateContentType tag. This tag prints out the summary of results appearing at the top of a results display. New content types would have to be understood by this tag.

3. Changing ArcCatalog. ArcCatalog would have to be aware of the new content types and recognize them for searching and publishing purposes.

4. Changing the Metadata Server component of ArcIMS. New content types would need to be recognized by the Metadata Server and included in the list of supported content types.

The only "safe" modification to the list of content types is an internationalization of the content type strings. All the strings are referenced by a numerical code, and it is only during display in Metadata Explorer that the numbers are translated to strings.

Alter the content theme dropdown list

Similar to changing the list of content types, changing the list of content themes is not recommended. The content themes used within Metadata Explorer are taken from the ISO metadata standard list of accepted data themes. Modifying the list of data themes would involve changes to more than just the Metadata Explorer application.

Changing the data themes involves:

1. Changing Metadata Explorer. The files that reference the list data themes are:

   - *search_form.jsp*—lines 131 through 153 (lines 152 through 174 in the advanced sample) print out the data themes in a dropdown list.

   - *search_form_advanced.jsp*—lines 200 through 223 (a newly created file for the advanced sample) print out the data themes in a dropdown list.

   - *res.properties*—lines 179 through 197 (lines 210 through 228 in the advanced sample) provide the mapping from the numerical data theme codes to strings.

2. Changing ArcCatalog. ArcCatalog would have to be aware of the new data themes and recognize them for searching and publishing purposes.

3. Changing the Metadata Server component of ArcIMS. New data themes would need to be recognized by the Metadata Server and included in the list of supported data themes.

The only "safe" modification to the list of data themes is an internationalization of the data theme strings. All the strings are referenced by a numerical code as defined in the ISO metadata standard, and it is only during display in the Metadata Explorer that the numbers are translated to strings.

Change the default search map
The search map that is displayed when Metadata Explorer is in search mode is an ArcIMS Image Service. To change the default search map:

1. Create a configuration file (*.axl) for the custom map. You can use Author or manually create a file using ArcXML.

2. In Administrator, Manager, or Service Administrator, create an Image Service using the configuration file from Step 1.

3. Add the Image Service to your aimsacl.xml file as an unrestricted or freely available service.

4. In aimsmeta.properties (/WEB-INF/classes directory), set the search_service_name to match the name of the service created in Step 2.

5. If the Image Service created in Step 2 is stored on a different server than the one that Metadata Explorer was installed on, set the search_host_name to the name of the server hosting the service.

6. If necessary, change the search_port_number to match the port number of the server hosting the service.

For a working example of a custom search map, the Advanced Metadata Explorer sample described earlier in the chapter uses a basemap of Canada instead of the default world map.

Enable gazetteer searches
In a default installation of ArcIMS and Metadata Explorer, gazetteer data is not installed and gazetteer searching is disabled in the search interface. Details for enabling the gazetteer functionality, including loading the data and setting up the necessary service, are provided in Appendix A of *Creating and Using Metadata Services*.

Changing the look of a map display

Change the map display layout
The map display layout is built as a series of subframes of the parent content_frame, as described in 'Metadata Explorer UI Organization' in Chapter 1. Lines 35 to 41 of content_frames.jsp build the map display frameset. Make modifications to the code between these lines to alter the map display layout.

Change the appearance of map components
The map display can be broken down into three components—map title, map legend, and the map itself. Each component appears in a frame of its own and is coded in its own JSP file. This ensures that modifying the appearance of one component will not affect the appearance of the other two.

Modifying the map title
All modifications to the map title can be made in map_title.jsp. Use HTML code to make visual display changes such as adding a background color to the title. Change the metadata.css file to make font style changes. The map title uses the style toolbarTitle defined in line 110 of metadata.css

Modifying the map legend
All modifications to the map legend can be made in map_legend.jsp.

The map legend is an image that is passed into map_legend.jsp from map_view.jsp. Use HTML code to make visual display

changes such as adding a background color behind the legend image.

Modifying the map view
All modifications to the map view can be made in map_view.jsp.

1. Lines 70 through 149 first retrieve several map parameters such as the map service name, map extent, and map units.

2. Lines 157 through 186 get a handle on the map's metadata document and retrieve the values of certain document properties and metadata elements using the <aims:datasetProperty> and <aims:loadElements>/<aims:getElement> tags. The values are stored as variables on the page.

   To retrieve additional properties from the existing ones, use either of the above-mentioned tags to retrieve the dataset property or metadata elements of interest. For details on retrieving information using these two tags, see 'Include different metadata elements' earlier in this chapter.

3. At this point, the code is broken into two sections:

   a. The code on lines 205 through 385 is for maps that are Image Services. The code makes an HTTP connection to the server on line 217 and obtains a handle to the map service on lines 226 through 246.

      Line 278 generates the map image, creating acetate layers for different map objects such as the scalebar or copyright text.

      To add additional acetate layers to the map, use the <aims:acetateObject> tag. Details on how to use this tag are available in *Customizing ArcIMS—Java Connector.*

   b. The code on lines 387 through 444 is for maps that are WMS Services.

      Lines 432 through 438 set the map extent, and line 441 retrieves the URL for the map image.

4. After the appropriate map objects for the map service type are created or retrieved, the code draws the map on the screen:

   a. Lines 538 through 612 draw the map tool buttons in their appropriate beginning states. Change the code between these lines to alter the map tools that appear in the map display.

   b. Lines 643 through 680 create a layer for the map and draw it to the screen.

5. When printing the map, map_print.jsp resends the map request to the server and retrieves the map and all associated objects anew. Any changes made in the above steps must be reflected in map_print.jsp.

Miscellaneous changes

Change the print page
Whether printing the quick view results of a search or browse, the detailed view of a dataset's metadata, or a map image, print.jsp or map_print.jsp always resend the request to the server, retrieving and displaying the results anew. As

mentioned previously, modifications to code that retrieve the results onscreen must be reflected in print.jsp and map_print.jsp.

Despite this, there are still some minor differences between the results that appear onscreen and the ones that are printed. These are:

1. For browse or search requests, results onscreen are shown in batches of a certain size, as defined by the aimsmeta.properties property batchSize. When printing results, however, the full result set is printed.

   To only print the current batch, modify the <aims:search> or <aims:browse> tags in print.jsp page to include the startResult, batchSize, and startBatchAt attributes. For details on these attributes, refer to *Customizing ArcIMS—Java Connector*.

2. For browse or search requests, buttons that appear below each dataset, such as the View Details or View Map buttons, are not printed, as they would not serve any purpose on a printed page.

   To include these buttons in the print page, remove the if statement

   ```
   <% if (!printPage) { %>
   ```

   on line 346 in data_output_include.jsp and its closing bracket `<% } %>` on line 383.

3. When printing the details view for a metadata document, the View Map button does not appear at the bottom of the results.

   To include this button in the print page, remove the if statement

   ```
   <% if (!printPage) { %>
   ```

   on line 958 (line 962 in the advanced sample) in details_output_include.jsp and its closing bracket `<% } %>` on line 1005 (line 1009 in the advanced sample).

4. On line 84 of print.jsp or line 310 of map_print.jsp, the header graphic file for print pages is displayed. This is the same header that appears on Metadata Explorer, with the additional text "Access a World of Information" on the right-hand side. To remove this text, either modify the graphic file meheader_lg.gif, or modify the print page to display me_title.gif instead.

Create custom error handling

In Metadata Explorer, the first line of JSP files that have the potential to throw Java exceptions reference the Metadata Explorer error page—error_handler.jsp. When an error preventing the execution of the remaining code occurs, the page location is redirected to error_handler.jsp.

The following Metadata Explorer files make a reference to error_handler.jsp:

- browse.jsp
- browse_results.jsp
- browse_results_predefined.jsp—a newly created file for the advanced sample
- data_output_include.jsp
- details_output_include.jsp

- find_place.jsp—attaches a parameter to the error page URL specifying that the source of the error is the gazetteer
- login.jsp
- map_view.jsp—attaches a parameter to the error page URL specifying that the source of the error is a map
- print.jsp
- search_form.jsp
- search_form_advanced.jsp—a newly created file for the advanced sample
- search_map.jsp
- search_results.jsp

By default, error_handler.jsp prints out a generic error message telling the user that an error has occurred. It is possible, however, to pass parameters to the error page and use the value of the parameters to customize the error page with specific messages.

For an example of parameters being passed to the error page, refer to line 1 in find_place.jsp and map_view.jsp. The parameters are included as part of the URL of the error page attribute on this line. For find_place.jsp and map_view.jsp, a parameter named 'source' is defined.

Modify the error page attribute on line 1 of any of the above-mentioned files to include custom parameters that will be sent to the error page.

All modifications to the error page itself can be made to error_handler.jsp:

1. Lines 16 through 22 retrieve any parameter values that may have been passed into the page.

   Modify the code between these lines to retrieve the values of any custom parameters you may have defined.

2. Lines 42 through 56 print out the error message to the screen. Modify the code between these lines to print out custom error messages depending on the value of custom parameters read from Step 1.