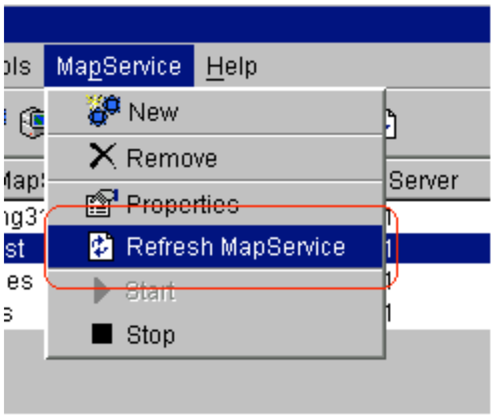
ArcIMS Template Version 2.0
Developers' Documentation

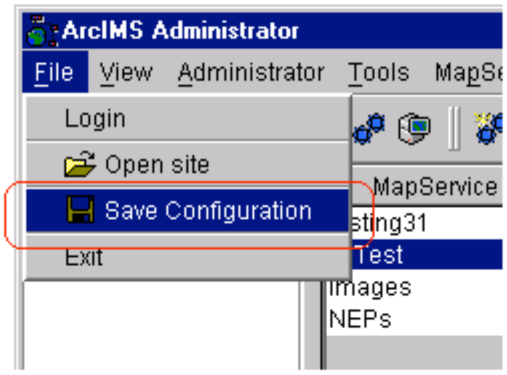
Creating an ArcIMS Application

There are a minimum of four steps in creating an ArcIMS application with the Center ArcIMS Template (referred to as the "Template" for the remainder of this document), the first two of which mirror exactly what you'd do to create a standard "out of the box" ArcIMS site:

1. Create an ArcXML file that lists the data to be displayed
2. Create an ArcIMS MapService based on your ArcXML file.
3. Make a copy of the ArcIMS Application Template in the C:\ArcIMS\Website directory (or any other web-enabled location on the computer) of the ArcIMS server.
4. Modify the "[application folder name]\includes\handleTools.asp" file to connect to the MapService you have just created.

The first two steps are dealt with in length in other documents. In brief, you will need to understand two applications, the ArcIMS Author and ArcIMS Administrator. The Author application will help you create your ArcXML file. This file can be further customized in a text editor. The Administrator application creates ArcIMS MapServices based on ArcXML files. This MapService must be created before ArcIMS can serve up your application. Please remember that you must have a login and password to use the Administrator utility. To receive a login and password, please see a member of the Template development team.





One hint on using the Administrator application. After each change you make to a MapService through either Author or Administrator, you will usually need to select "Refresh MapService" from the "MapService" menu. You must then also select "Save Configuration" from the "File" menu to ensure your changes will happen in the future. Until the "Save Configuration" menu item is selected, no change is permanent.

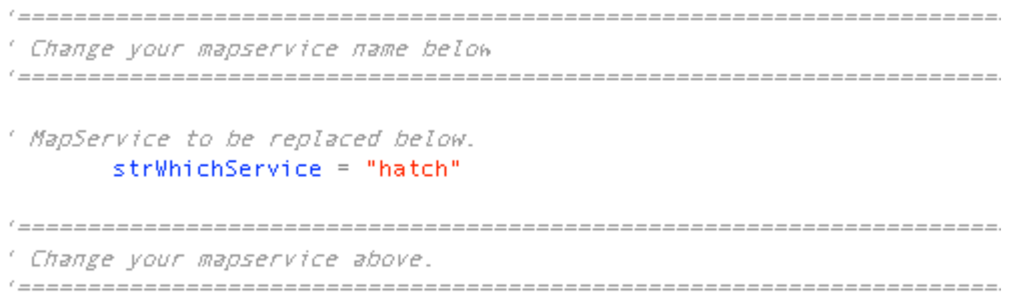
With the release of version 2.0 of the ArcIMS Template, there is also an Avenue script that will take each view in an ArcView 3.x project and create an ArcXML (or "axl") file that contains much of the information that was stored in the ArcView project. This script is stored at the same level as this document, and is named, "apr2axlSepViews_latest.ave".

Copying the Template to Your New Directory

Copying the Template to the ArcIMS project's directory can occur in two ways. It is perhaps easiest to simply right click on the "version1" folder (located at the following location: \\test-ims\ArcIMS\Website\version1) and select "Copy". You can then right click in the Website directory and select "Paste". This will create a copy of the template with all of its files and folders intact. You will then want to right-click the folder you just pasted into the Website directory and rename it.

Alternately, you may want to use the Visual Basic (VB) tool that copies the template for you. Please contact Ruffin Bailey if you would like to use the VB utility.

Finally, you must modify the "handleTools.asp" file in your newly created and renamed directory so that your ArcIMS application will point towards the correct MapService that you created in author. To do this, open the file in a text editor (*e.g.*, UltraEdit) and find this section in the code.



```
'======================================================================
' Change your mapservice name below
'======================================================================

' MapService to be replaced below.
        strWhichService = "hatch"

'======================================================================
' Change your mapservice above.
'======================================================================
```

Change the text between the quotes to the name of your MapService. This can also be done in the VB tool mentioned above.

Save the file and your ArcIMS Application is ready to go.

Other Set-Up Items of Note

There are a few objects in the handleTools.asp file's Init(strService) subroutine under the heading, "It's Make Eric Treml happy time!" In this subroutine, the map's North Arrow, scale bar, and labels are created. Please check in this location to see how to modify color, size, location, and other properties of these objects.

Finally, please remember that this version of the template is supported only in Internet Explorer 5.0+ on Windows, Macintosh, and Solaris operating systems. Mozilla and

Netscape 6.0+ can be used to view the template, and bug reports from these browsers are being taken at this time, however there are a few known bugs with the Template and these browsers as well as a few instances where full functionality is not available.

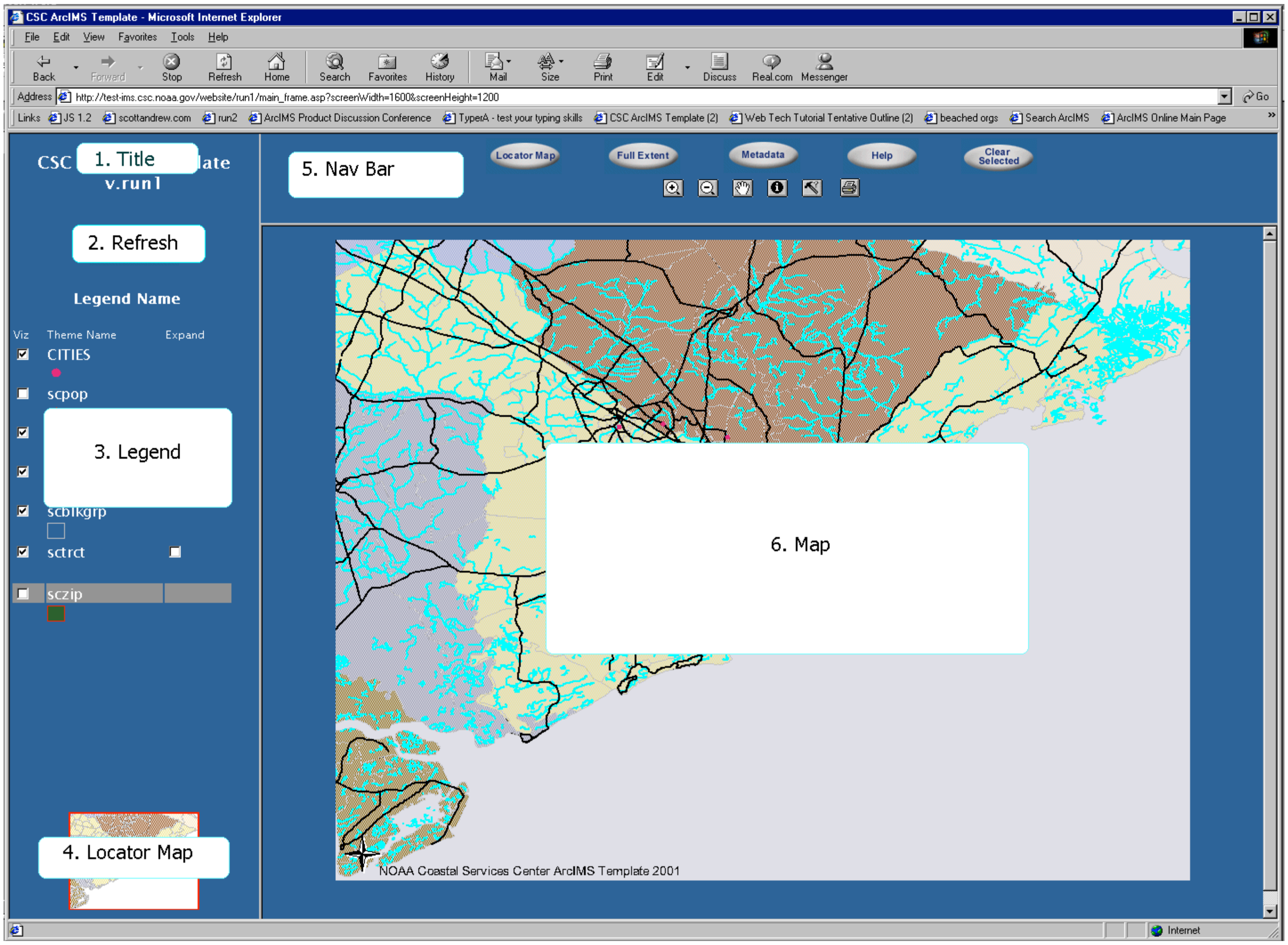
How the Template Works (for Developers Only!)

The template is written using a number of relatively advanced, html-related technologies including javascript, Cascading Style Sheets (CSS1), Document Object Model 1 (DOM1) and a number of browser-specific "tricks". Because of the high level of functionality in the template, it can only be used in Netscape 6, Mozilla, or, preferably for version 2.0, Internet Explorer 5.0+. It is recommended that ArcIMS applications with a larger intended audience also create a no-frills ArcIMS javascript/html viewer application with the ESRI's ArcIMS Designer application. Browsers that can view this viewer but are not able to access the Template can be shuttled off to the html viewer created by Designer automatically via javascript.

On the server, the Template uses Microsoft's Internet Information Server (IIS) with Active Server Pages (ASP) to dynamically create the images and web pages sent back to the Template's end users. ASP uses ArcIMS's Active X connector to communicate with the ArcIMS server. Information is exchanged between the Template and ArcIMS and is then packaged for IIS to send as a web page back to the end user.

To edit or customize the Template directly will require that you know both the client-side technologies above and Visual Basic Script, which is used by ASP.

The Parts of the ArcIMS Template



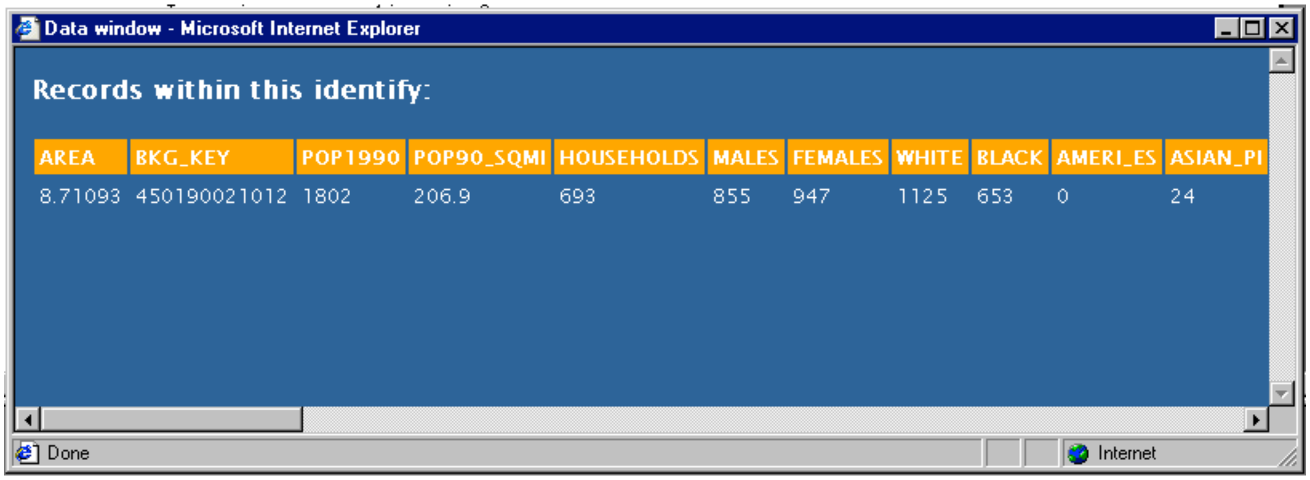
The Template is split into six frames as listed above. The main frameset can be found in the file "main_frame.asp". Below are the frames listed by name, by file, and by the Javascript name used in main_frame.asp.

Frame Name	**File Name**	**Frame Name in main_frame.asp**
Title	title.htm	title
Refresh	refreshLayers.html	reLayers
Legend	legend.asp	legend
Locator Map	locate_map.asp	locate
Nav Bar	navigation.asp	Toolbar
Map	map.asp	Map

The functions performed by these frames will be dealt with under a separate heading.

Child Windows

There are also two child windows that appear when needed. The first is the window for displaying data returned by an identify performed on the map. This file is named "data.asp" and is found in the childWin directory, on the same level as the frame source described above.

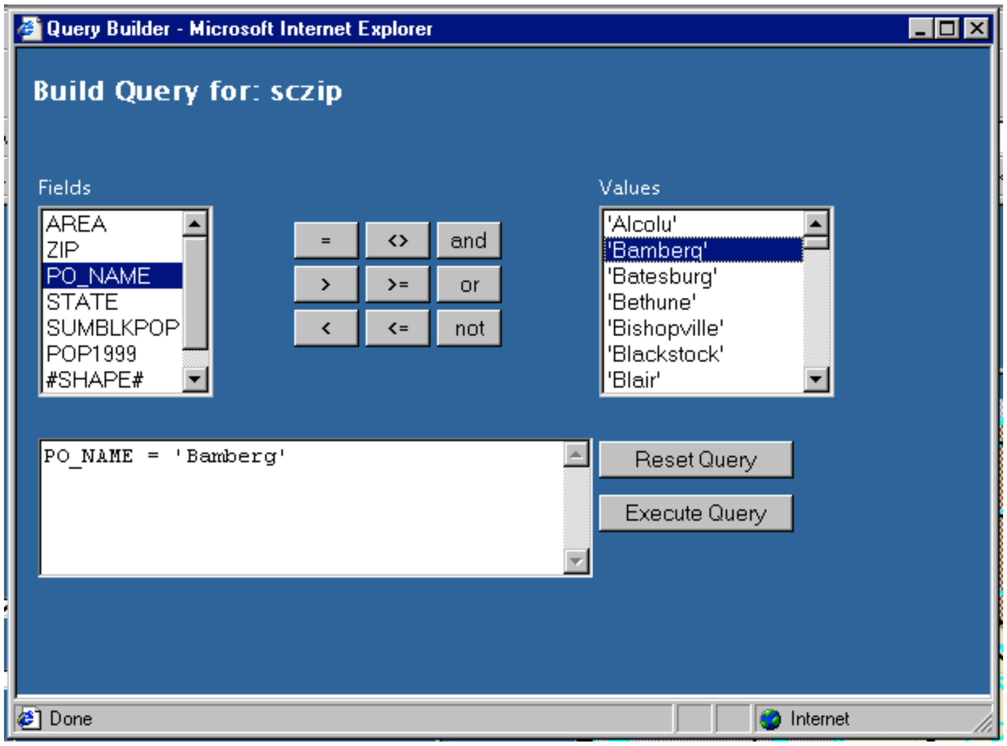


Records within this identify:

AREA	BKG_KEY	POP1990	POP90_SQMI	HOUSEHOLDS	MALES	FEMALES	WHITE	BLACK	AMERI_ES	ASIAN_PI
8.71093	450190021012	1802	206.9	693	855	947	1125	653	0	24

The data window can be customized to reduce the number of fields displayed. Colors can also be changed if needed.

The Query Window is another important child window. It can also be found in the childWin directory of the Template application. It consists of two frames, one of which is displayed to the user and the other is hidden to allow efficient variable processing. The frameset is stored in the "queryFrames.asp" file, the visible frame in "query.asp", and the hidden variable frame in "queryVars.asp".



```
PO_NAME = 'Bamberg'
```

When a user clicks on a new field from the Fields select box, the query.asp window uses javascript to perform a number of tasks. If the field has not been selected before, the queryVars.asp page is reloaded and passed a variable defining which field has been requested. The queryVars.asp frame then retrieves all possible values of the field from the ArcIMS server and displays them in the Values select box. If there are more than 250 unique values, only the first 250 are displayed in the default Template and the end user is alerted.

These new values are also stored in the query.asp frame. If the query window is not closed and a field is re-selected, queryVars.asp is not reloaded and the values from the initial retrieval is reused, saving the end user time.

Note that the query window will close after alerting the user if an unqueriable layer (*e.g.,* an image) has been selected.

Breakdown of Template Frames in Main Template Window

1. title.htm

Very simple, straight html frame. Intended to be used simply to provide a link from the mapping application back to the parent web site.

2. refreshLayers.html

Mostly unremarkable other than that its single image is used by other frames to determine if a new map is being loaded. Clicking the image will typically refresh the layers in the map by submitting the mapForm in map.asp. Once a new map requested, the "button" image should be replaced by an image that gives a visual cue that data is being loaded. Other frames then examine the src property of this sole, unnamed image to see if a new map is being pulled, and, if it is, to know to lock out other commands.

3. legend.asp

Javascript Functions:

- setTool(strToolName) - Sets current_tool in mapForm of map.asp to strToolName.
- rememberHidden(strWhichLayer) - not sure this is used any longer.
- changeViz(strName, intIndex) - Changes mapForm's "visible_[strName]" to 1 or 0, whichever it isn't at the time.
- checkLoad() - Returns true if the Template is not retrieving a new map and false if a new map is in the process of loading. Used to lock out commands during map reloads. Is also repeated in a number of frames.

- collLay(strName, intIndex) - "Collapse Layers" - Checks to see if we're able to collapse a layer's compound renderer display in this frame or if we're loading a new map and should lock the request out.
- forceLayer(strName, intIndex) - if we had a layer's info expanded before, we'd want to have it expanded again when we redraw the layer. Sometimes the layer's done before we've finished retrieving a new map, we don't want to lock out this legend initialization. CollLay actually calls this function if we aren't loading a new map; this function just works whether we're loading a new map or not.
- makeActive(hiMe, intThemeIndex) - turns the background behind the layer title in "hiMe" ("highlight me") to gray, showing it's the active layer. Runs a check to make sure that a new map isn't being received. If we're getting a new map, this can't be done. intThemeIndex is used to set activeTheme in mapForm so that it knows which layer is active when a tool is selected.
- forceActive(hiMe) - called by makeActive if checkLoad() returns true. Also called to initialize the way the legend looks initially when it's created/rebuilt.

Html Forms

- legendForm (note that this form is never actually "submitted")
  - visible_[layerName] - checkbox, one for each layer in the map - when clicked, calls changeViz(layerName, themeIndex)
  - chk[layerName] - checkbox, one for each compound renderer - when click, expands the view for the legend.

4. locate_map.asp

HTML Forms
- locMapForm -- currently disabled
  - [nameless] - image submit - currently not used, but will later be enabled to allow panning by clicking on the locator map. The IMAGE's src is taken from Session("locLocation"), which is set in map.asp.

5. navigation.asp

Javascript Functions

- setTool(strToolName) - map.asp's mapForm "currentTool" is set to be strToolName. mapForm is not submitted, however.
- setTools2(strToolName, imgName, strNewImg, strOldImage) - this function does the same as setTool, but also changes an image's src and resets the last changed image's src to what it was before. Used specifically to have only one button look depressed from the "persistent tools" -- i.e., this is used to set current_tool to "zoom_in", then change the zoom in button to the "depressed" look, and then "undepressed" pan, zoom out, or whatever was last.

- flipLocMap() - as in "flip it on, flip it off" - Toggles the locator map off/on. Fully implemented in IE5+. In NS and Mozilla, it would take a complete reload of the main frameset to "erase" the locMap frame, so the image is only blanked out. This is really nearly pointless, and a better fix should be found.
- checkLoad() - Returns true if the Template is not retrieving a new map and false if a new map is in the process of loading. Used to lock out commands during map reloads. Is also repeated in a number of frames.
- openQuery() - Opens the query builder window if checkLoad() returns true.

6. Map.asp

ASP Includes:
- writeRQS.asp - used to display all variables in the QueryString for debugging. Enabled by setting variable verboseDEBUGGING to true. Not discussed further.
- handleTools.asp - This include handles all of the communication with the ArcIMS server and the main Template window. Explained in more detail below.

ASP Functions:
- removeDups(tempString) - Used on hidden variable strings to prevent overly long variables.
- getLegend_RefreshMap - Refreshes the map and was originally used to create an image legend.

Javascript Functions:
- warnOff() - uses the innerHTML property of a span element to erase its contents. Specifically used in this case to remove the "The map is currently loading. Please be patient." warning when the template refreshes or produces a new map.
- printMe() - used to print the map.asp frame, but is not utilized in this version of the Template.
- findSelected() - Another function I need to remove.

Javascript Includes:
- bounding.js - listens to mouse events and creates the bounding box when the zoom in tool is enabled. Currently the zoom out tool does not employ a bounding box.

HTML Forms and Form Elements:

- mapForm
  - queryString - hidden - This is filled in by the query child window when the "Execute Query" button is pressed. It is used in the "query.asp" include in handleTools.asp. And if that wasn't enough, you reference it with the code "Request.QueryString("queryString")".
  - queryLayer - hidden - Tells query.asp in the longTools directory which layer should be queried. Only filled in (and only used) via the query child window.

- activeTheme - hidden - this holds the index of the theme that should be considered active when we run an ArcIMS request.
- lastPersistent - hidden - There are a number of "one-shot" tools that can be invoked by the Template. It is important to remember which tool was selected before the one-shot function was invoked. For instance, if the "zoom" button is depressed in the navigation frame and the user then presses "Full Extent", once the full extent has been done, the Template must remember that the zoom button is depressed in case the user clicks on the map, now at full extent.
- current_tool - hidden - This variable provides a string to the handleTools.asp include so that it can determine which function to perform on the map. Set by many windows.
- collapse_[layerName] - hidden - This holds either a zero or a one, depending on whether a layer should be collapsed or expanded in the legend. Compound renderers can be expanded to show each value in the legend, and when map.asp is reloaded and legend.asp's code recreated, this state must be maintained. A zero indicated not expanded, a one for expanded. Backwards, I know.
- visible_[layerName] - hidden - remembers which layers should be visible in the map, and also which boxes should be checked in the legend.
- screenHeight - hidden - this variable is written out using javascript into the page in order to remember the resolution of the user's screen when they refresh the map after their Session has expired.
- screenWidth - hidden - this variable is written out using javascript into the page in order to remember the resolution of the user's screen when they refresh the map after their Session has expired.
- ~~[unnamed] - image submit - This is the actual map image. It is an image type input in order to capture the location of user clicks within the map.~~ This image is now a standard html image tag. It is, however, now included within a div tag to allow for the use of a modified version of ESRI's bounding box javascript code. There are also several form variables moved to this location to ensure that they are displayed correctly when debugging flags are turned on.

<u>bounding.js</u>

This is a file taken from ESRI's ArcIMS downloads site and allows the bounding box to be drawn on ArcIMS map images. Major parts of the code and some parts that were customized are listed below.

Javascript Functions

- startUp() - called by map.asp's body's onLoad event. This function begins the page's listening to mouse events (clicks and movements) as well as the functions that will process the input from "what's heard" from these events.
- startZoomBox(e) - starts the display of the bounding box, and sets up the page to redraw the box as the user mouses around the image. Also remembers the x,y coordinates of the first mouse click.

- mapTool (e) - makes sure that the user has the Zoom In tool currently selected before drawing the bounding box. In other words, if the Zoom In tool isn't selected and the user clicks in the map frame, this function ensures that the bounding box isn't drawn.
- chkMouseUp(e) - easily the most modified function from the original ESRI file. This function fires on user events trapped from the page. First, this function stops the bounding box from being displayed if it was currently visible. After this, it determines whether the tool being used is the Zoom In tool. If so, it propagates the x,y (stored during the firing of startZoomBox function), x2, and y2 variables in map.asp's mapForm and submits mapForm. If not, it takes the current mouse coordinates, places them in x and y, and submits mapForm.

handleTools.asp (included in map.asp)

ASP Includes
- skinnify.asp - prepares a locator map from the main map service. Not discussed further in this document.
- newLegend.asp - Creates the code for the dynamic legend in the Legend frame.
- longTools/query.asp - Handles the query function and adds a yellow layer to the map to show the results of the query. Not discussed further in this document.
- longTools/identify.asp - id's the clicked feature, if there is one, and pops open the data.asp window with the results of the id. Not discussed further in this document.

ASP Functions
- init() - this initializes a map based on connector information and places the map object in a session variable. Also initializes the locator map on the same connection information. Additionally, this is where the app sizes the image for the main map based on screen resolution.
- rAlert(strAlert) - this function "raises" an alert by writing out javascript to the Response object that will pop open an alert box with "strAlert" as its text contents.

newLegend.asp (included in handleTools.asp)

Currently only called when the Session expires and a map is initialized or when map is "initially initialized ". Not reloading legend each hit to increase performance.

ASP Includes

- handleRendererSub.asp - deals with stepping through the renderers of each layer. Called as a subroutine in this include.

ASP Functions

- rgbify(dblImsGlobalColor) - takes an IMS color and turns it into its respective rgb value for a stylesheet or what-have-you. The output is a string that has the "rgb" function included, *e.g.* "rgb(51,102,153)".

handleRendererSub.asp (included in newLegend.asp)

ASP Subroutines

handleRenderer(rendObj) - originally just a plain jane include, this was turned into a subroutine so that it could call itself recursively if a compound renderer appeared. This sub makes the strSecondRow variable by spidering through renderers and creating code to create a legend that matches the information the renderers represent. This is where half of the legend work is done.

handleFillStyle -- This function take an imsFillStyle integer and returns a string for the corresponding image. Note that the string is a relative path, *e.g.* src="./images/LegIcons/down_dia.gif" This is used to create hatch and lined patterns with polygon renderers.