Advanced MapServer User Interfaces

Stephen Lime

Data & Applications Manager

Minnesota DNR, MIS Bureau

Workshop Objectives

- hands-on experience in building a complex application
  - application components on-demand
  - runtime configuration via HTML forms
  - javascript/DHTML
  - coupling external tools with MapServer
- when to use CGI vs. MapScript, DHTML vs. Java
- exposure to a few version 4.0 features

Workshop Plan

- a basic application: Itasca County
- starting simple, multiple scalebars
- pan controls the easy way
- frames for query results
- rubber-band zoom using DHTML
- (time permitting) interfacing with helper apps:
  - gazetteer
  - authentication

Workshop Resources

- Itasca application installed at:

  `C:\Program Files\Apache Group\Apache2\htdocs\workshop`
- Itasca application URL is:
  `http://localhost/workshop/index.html`
- use *notepad/ConTEXT* to edit application files
- edit files ending in ".student.html"
- complete files (no cheating) are missing the "student" part
- there are AM and PM versions "_pm" or "_am"

Template Caveats

- [program]: name of the MapServer CGI binary
- [root]: location (relative to document root) of the Itasca application
- [map_web_imagepath]: location of the temporary image directory
- [map_web_imageurl]: name of that directory relative to document root

Basic Application

- our old friend the Itasca County, MN demo application
- check out itasca_basic.html *(please don't edit the file)*

start the application

Exercise 1: Multiple Scalebars

- information needed to make a scalebar
  - **relative** or **absolute** extent of a map
  - size of the map (in pixels)
  - unit of measure for the map coordinates
  - unit of measure for the scalebar
- change map file parameters on-the-fly using CGI variables

On-the-fly Configuration

- MapServer CGI allows for almost every aspect of the map file to be changed via HTML forms
- syntax is map_object_parameter=value (e.g. map_lakes_minscale=10000)
- for security reasons you must use the new DATAPATTERN or TEMPLATEPATTERN options to change templates or data files
- ability to define data (inline features)

Exercise 1: Tasks

- edit itasca_adds_scalebar.student.html
- add a second scalebar to the right of the existing scalebar that represents kilometers instead of miles

start the application

Exercise 1: Solution

One image tag src value that works is:

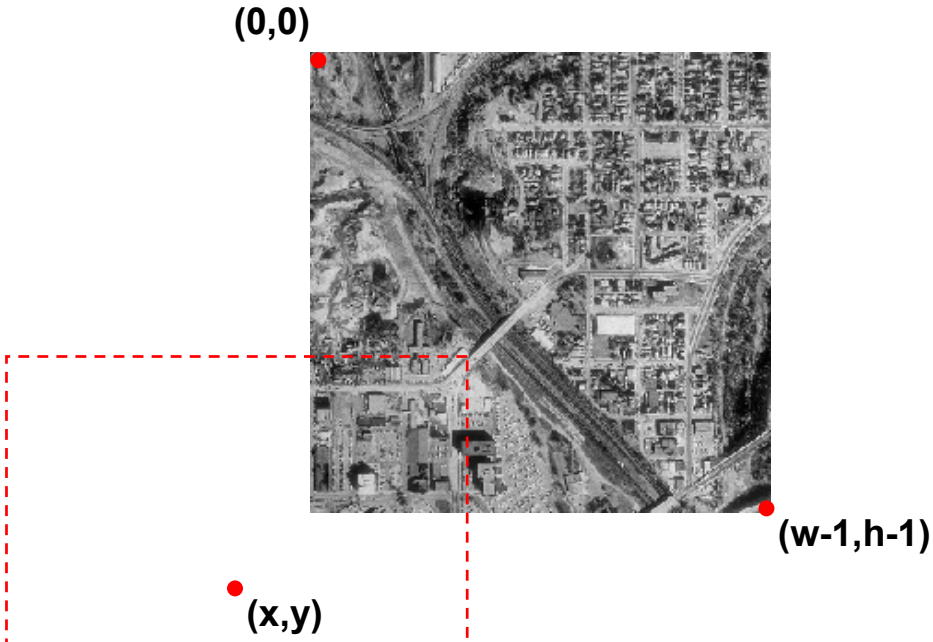
```
src="[program]?map=[map]&
mapext=[minx]+[miny]+[maxx]+[maxy]&
map_scalebar_units=kilometers&
mode=scalebar"
```

Exercise 2: Pan Arrows

- convenience feature that should be part of every application
- can be implemented using individual directional images or an image map
- leverages the notion of a virtual image

Exercise 2: Virtual Image

- MapServer will recognize “mouse” clicks outside of the real image



Exercise 2: Tasks

- edit itasca_adds_pan.student.html
- finish the Javascript pan() function that computes a MapServer URL representing a pan in a particular direction

start the application

Exercise 2: Solution

To pan to the southwest:

```
x = 0 - 600*pansize - 600/2;
y = (600-1) + 600*pansize - 600/2;
```

Exercise 3: Frames

- to make applications user-friendly it's convenient to direct query results someplace other than the main window
- 3 options: popup windows, layers or frames

Javascript, Windows & Frames

- windows are at the “root” of the javascript object hierarchy
- windows contain documents or frames
- frames can contain documents or other frames, and so on…
- documents contains elements like forms, images and so on…

Exercise 3: Tasks

- edit itasca_adds_frames.student.html
- Fill in the “submit_form()” function so that if in browse mode then the application uses the main frame and if in a query mode it uses the query_results frame

start the application

Exercise 3: Solution

Here's one possible code block:

```
function () {
  if(document.mapserv.mode[0] == checked) // browse
    document.mapserv.target = "_self";
  else
    document.mapserv.target = query_frame;
  return;
}
```

Exercise 4: Rubber-band Zoom

- rubber-band zooms are a terrific, intuitive way to spiff up an application
- users can quickly define an area-of-interest, saving server resources
- emulate the functionality of the DNR LandView interfaces (java/dhtml)
- utilize packaged coordinate management and DHTML component libraries

Coordinate Management

- Javascript library mapserv.js
- contains code to manage extents, a variety of zoom types and layers
- integrates with a couple of component pieces (that enable rubber-band zooms), these components are written in DHTML and Java

DHTML vs. Java Applet

- neither is a perfect solution
- personally I prefer the Applet: self-contained and extensible
- however, standards-based browsers and cross-browser scripting libraries make DHTML a viable alternative for simple needs

Impact of DHTML on Design

- problem: in order to know where you are within a DHTML layer you must use absolutely positioned layers, however this limits design.
- solution: use relatively positioned DHTML layers to “anchor” absolutely positions DHTML layers

example DHTML page

DHTML Rubber-band Map Control

- Javascript library dbox.js
- built on top of the CBE cross-browser DHTML library
- dBox objects leverage the anchor concept mentioned previously, so for every dBox there are 2 DHTML layers necessary to place the component on the page

mapserv.js & dbox.js integration

- glued together using some event-based javascript functions that you write
  - These include:
    - seterror_handler()
    - setbox_handler()
    - reset_handler()
    - mousemove_handler()
    - mouseexit_handler()
    - mouseenter_handler()

Exercise 4: Tasks

- step through the conversion of the Itasca application to DHTML
- you have a version you can “play” with, itasca_adds_dhtml.student.html
- extra credit: limit box zooming/query to appropriate form settings (e.g. no box when zooming out)

start the application

MapServer Integration: Gazetteer

- problem: you want to allow users to quickly do place-name lookups
- benefits: customer service, performance issues
- solution: use an external database and program to do the lookups (Perl/JSP/ASP/…)

Gazetteer Example

- Recreation Compass
- MySQL used to store USGS GNIS data, 28,000 places
- Perl script used to query MySQL
- MapServer URLs or calls to mapserv.js objects are used to reposition the application

MapServer Integration: Authentication

- problem: you've got data you'd like to password protect
- solution: depends on problem!
  - you could use http server URL patterns and normal web authentication
  - or you may want to leverage internal security of a particular data source like SDE or PostGIS
  - there are certainly other alternatives such as using application servers like WebLogic or Zope

Authentication Example

- DNR Heritage Data Viewer
- sensitive data stored in SDE, SDE/Oracle authentication required
- validation of credentials done using a small JSP page and SDE Java API
- username/password made available to MapServer using cookies and %variable% map file pre-processing

For more information…

Contact: Stephen Lime
steve.lime@dnr.state.mn.us