1. Migrate GeoSVG to Firefox
2. GDrawing

Xun Lai

October, 2005

Adobe SVG support vs. Mozilla native SVG support

- There is no significant difference. Both follow the W3C specifications quite strictly.
- Status of Mozilla SVG support
- Differences found so far during migrating GeoSVG to Mozilla

ASV	Mozilla
createElement(…) OK	Must use createElementNS(http://www.w3.org/2000/svg, …)
setNodeValue(…) OK	Must use nodeValue = …
Default width of characters are different	
	Text following a path has some problem
	<pattern> element has some problem
	<line> with stroke-dasharray style has some problem

Advantages of Mozilla SVG Support over ASV

- Written once and work on Windows, Mac, and Linux
- No more inter-document communications problem
- SVG tags can be mixed with XUL and XHTML tags while embeding an SVG file via <embed> is still supported.
- XUL GUI widgets can be used to enhance an SVG application.
- Mozilla is making progress every day.
  - Mozilla native SVG support is by default turned on in Firefox 1.5 before which the support was immature.
  - Time is right for us: 1.5 Alpha 1(May 31,2005), Alpha 2(July 12), Beta 1(Sept. 8), Beta 2(now available), 1.5 Final(2005)

Difficulties Developing GUI Widgets under ASV

- GUI widgets needed in GeoSVG: button, input box, checkbox, select, menu, toolbar, (modal) dialog box, tabbed dialog box, and maybe more.
- It’s possible to develop all these widgets, but it’s time consuming and also bug-proned.
- When a GUI widget, especially for those complicated GUIs, is not well designed, the codes for GUI and codes for the real task cannot be decouple easily.

Difficulties Developing GUI Widgets under ASV (cont.)

- Need to implement:
  - Draw the outlook of each widget
  - Mechanism for how to layout widgets.
  - Simulate modal dialog box
  - Simulate onfocus event of many widgets. It becomes more complicated when dialog box presents
  - Unlike a real menu, a simulated menu cannot stretch outside the window an SVG resides in
  - Simulate a flashing cursor in a simulated input box
  - Capture keystroke for input box (Some browsers on some OS do not pass the keystroke to the Adobe SVG Viewer plugin)

Tradeoff Approach to Develop GeoSVG

- The GeoSVG authoring environment will be developed to only work under Firefox.
  - It will only use those SVG features that also work under ASV so that it’s possible to develop one version for ASV in the future if a full set of GUI widgets is ready for ASV
- Drawings (manipulatives) created by the authoring environment can work mostly under both Firefox and ASV. Drawings needing keystrokes are the exceptions.

The Partially Migrated GeoSVG

- The partially migrated GeoSVG using Firefox native SVG support, XUL and XBL
  - It looks like a standalone application.
  - The object property dialog box
  - The calculator to define relations among objects
  - XBL is very helpful for defining property dialog box for different objects. It simplifies a lot of codes.

GDrawing Library

- http://wme.cs.kent.edu/geosvg/documentation.html
- Terminology
  - Main Program:
    - topic lesson editing page
    - an assessment question editing page
    - a bulletin board message composing page
  - Drawing Window: a pop-up window containing the GeoSVG authoring environment

GDrawing Library (cont.)

- Simple APIs:
  - GDrawing.newDrawing(id, parentId)
  - GDrawing.editDrawing(id)
  - GDrawing.removeDrawing(id)
  - GDrawing.loadDrawing(id, parentId, str, width, height)
  - GDrawing.changeDrawing(id, str, width, height)
  - GDrawing.getDataStr(id)
  - GDrawing.getWidth(id)
  - GDrawing.getHeight(id)

GDrawing Library (cont.)

- Easy use for main program
  - no need to deal with something like inter-window communications, inter-document communications
  - If the main program has a pre-defined insertion point for a drawing, it doesn't need to deal with createElement, appendChild.
  - If the insertion point such as a <p> or <span> element for a drawing is dynamically generated, the main program only needs to do createElement and appendChild operations for the insertion point element.
  - It's the main program's job to save the data string, width, height of the drawing back to the server.