Faculty of Science
Department of Computing

ITEC830: XML Technologies

Report for Assignment One

Semester 2, 2012

Student Name: Shamim Ahmed

Student ID: 42756561

A Brief Introduction to XSLT

XSL Transformation (XSLT) is one of the most widely used technologies related to XML. Formally speaking, XSLT is a part of XSL (Extensible Stylesheet Language), which has two other components: XPath and XSL-FO. XSLT is a W3C recommendation.

XSLT can be considered as a language for transforming input XML documents into other XML documents, or documents with different formats like HTML or text. One key point to note is that an XSLT stylesheet itself is written in XML. This allows XSLT to benefit from many XML-related features like namespace and validation.

One of the most common uses of XSLT is to translate an XML document into an HTML document. This is because XML is suitable for data storage and transmission, but it is not designed for data presentation.

XSLT is declarative, which means that it focuses on specifying the problem, rather than specifying how to obtain a solution step-by-step. In this regard, XSLT shares many of the characteristics of functional programming languages.

The basic building block of an XSLT stylesheet is a template, which is identified by an <xsl:template> element. A template may be invoked when a particular element in the input XML tree is encountered. Within the template body, the programmer can define the processing steps for the matched element. This may require printing some HTML/text, or invoking other templates.

XSLT has constructs for conditional execution of statements and looping. It is possible to load multiple XML documents and process them in a single XSLT stylesheet. On the other hand, an XML file can contain a reference to an XSLT stylesheet, which indicates to an application (e.g., a browser) that the XML file should first be transformed by applying the stylesheet. The application can display the output of the transformation process.

XSLT uses XPath to select subsets of input XML tree for processing. This also means that various XPath functions (e.g., 'position ()') can be used within an XSLT stylesheet.

Implementation of Extensions

I have implemented the following extensions as part of the assignment:

- I've added icons with forecast information whenever possible.
- I have made the tabular data sortable.
- I've integrated Google map support.

Weather Icon

It is a standard practice to show weather forecast with icons. I've downloaded the icons from the Bureau of Meteorology website (http://www.bom.gov.au). Please note that a weather icon is displayed only when the relevant information (forecast icon codes) can be extracted from the input XML files.

Two variants of the same weather icons have been used in the HTML page: large and small. It is possible to collect both sets of icons from the Bureau of Meteorology website. The website also provides documentation that specifies which icon should be used with which forecast code (Australian Bureau of Meteorology 2012).

Sort option for tabular data

Most of the forecast data has been presented in tabular format in the output HTML page. In order to enable users to interact with the data, we have made such tables sortable. For tables that display forecast information, there are three sort options:

- Sort by location name
- Sort by minimum temperature
- Sort by maximum temperature

The table that displays event information can be sorted only by event titles.

We have used JQuery table sorter plug-in to accomplish this. This plug-in is flexible and it allows the user to specify which columns should be sortable. It also shows icons in sortable columns in order to indicate the current sort direction.

Google map support

Many of the locations appear as a link in the HTML file. If the user clicks on such a link, a new tab opens which shows the location marked in a Google map. By clicking on the marker, the user can see the location name, along with the latitude and longitude.

In order to implement this feature, I have used Google map API version 3 (Google Inc. 2012). It is necessary to know the latitude and longitude of the location before the map is rendered via JavaScript. This is done with the help of REST-based map web service, which returns the location information in XML.

Due to the 'same origin policy' (Wikipedia 2012), we cannot use Ajax to communicate with the map web service from the HTML page. As a result, we have written a Python script to handle the communication with the web service. The flow of information is thus the following:

- The JavaScript code in the HTML page submits an Ajax request to the python script. As part of this request, the name of the location is specified.
- The python script communicates with the external map web service, parses its XML response and returns only the latitude and longitude as part of the result.
- The JavaScript code in the HTML page uses the latitude and longitude to render the Google map, along with a marker.

Other third-party libraries

- The python library 'lxml' has been used for parsing the XML received from Google map service.

- JQuery, a popular JavaScript framework, has been used for most JavaScript related functionalities (like Ajax request submission).

Issues faced during Implementation

The following issues were faced during the implementation:

- It required some time to figure out how to establish the relationship between different areas using their 'aac' and 'parent-aac' attributes.
- Since most browsers do not support XSLT 2.0, I had to depend on XSLT 1.0 features. This made the task a bit difficult in some cases. For example, XPath 2.0 has a function 'format-date()' which can return day of week for a given date. This function is not available in XPath 1.0, so I had to implement this feature. The relevant code was partly taken from the book 'XSLT Cookbook' (Mangano 2005).
- Integration with Google map was not straightforward, due to 'same origin policy'. Finally I wrote a python script as a workaround.
- If we pass only the name of the city/town to Map web service, then it may return multiple matching locations (it happens for 'Hamilton', for example). In order to avoid this problem, the python script appends the country code (AU) with the location name before it is sent to the web service. For the Victorian cities, it also appends the state name ('Victoria'). This returns unique results for all the cities and towns that we need to show in our HTML page.

References

Australian Bureau of Meteorology 2012, *Icon Descriptions – Next Generation Forecast Services*, viewed September 22, 2012 <http://www.bom.gov.au/NexGenFWS/icontable.shtml>

Google Inc. 2012, *V3: The Solution for Maps Applications for both the Desktop and Mobile Devices*, viewed September 22, 2012 <https://developers.google.com/maps/documentation/javascript>

Wikipedia 2012, *Same Origin Policy*, viewed September 22, 2012 <http://en.wikipedia.org/wiki/Same_origin_policy>

Mangano, S 2005, *XSLT Cookbook, 2nd Edition*, O'reilly Media, Inc., pp. 114-115.