Week 6

Google Maps API

Google has a library of JavaScript functions that allow our programs to load maps of all sorts into our documents. GDownloadUrl() comes from this same library, but most of the other classes and functions are for using their maps. You can read about this library here:
http://www.google.com/apis/maps/documentation/index.html

In the next section, you will find the hello world of Google maps, with filename "googleMapTest.html". The script tag at the top of the file contains the key for ctislab, so it may not work on your computer. Please substitute your own key to be run this program and view the map.

Notes on googleMapTest.html, function loadMap():
- The constructor GMap2() takes as parameter a reference to the div you want to be surrounding the map in your document.
- The first parameter to setCenter() is the GLatLng object that specifies where you want the map to be centered. A GLatLng object contains the latitude and longitude, as is customary in cartography.
- the second parameter to setCenter() is the zoom level. A zoom level of 0 shows the whole world map, and a zoom level of 20 shows approximately one residential lot.

More on how to center the map in the next section...

googleMapsTest.html

Instructions: Shows how to display a map on the browser window, assuming you know the latitude and longitude of the place you want at the center of the map.

googleMapTest.html

What do you want to map?

In order to set the center of a map, you need a GLatLng object to send as a parameter to setCenter(). There are two ways you can get a GLatLng object:

1) As in googleMapTest.html, if *you know the latitude and longitude*, you can send them to the GLatLng constructor:

```
var glatlng = new GLatLng(37.4419, -122.1419);
map.setCenter(glatlng, 13);
```

2) If you *need to lookup the latitude and longitude*, you need to make a request to the Google server and get a response from your callback function:

```
function getMap()
{
   var geocoder = new GClientGeocoder();
   geocoder.getLatLng("12345 El Monte Road, Los Altos Hills, CA", createMap);
}

function createMap(glatlng)
{  var map = new GMap2(document.getElementById("map"));
   map.setCenter(glatlng, 13);     // called by the geocoder
}                                  // when the response is ready
```

The process of turning an address or a country's name into a geographic point is called "geocoding." You can see the whole program and run it by downloading "geocoding.html" from the next section.

geocoding.html

Instructions: Demonstrates how to make a map from an address or city name.

geocoding.html

Dojo

Dojo is a popular toolkit that allows us to use it as a simple code libary. Dojo code is degradable, cross-platform, and open source. In this section, we will see how easy it is to put an editor onto the document window using Dojo.

Instead of having to download all the Dojo code libraries onto our local computers, Google allows our web page to load the Dojo code we need for our web application from its servers at runtime. Google is acting as a "Content Delivery Network" here, and somehow they get around the rules against Cross Site Scripting.

Loading the code from Google's servers may take some time, though, so we make an asynchronous call to a function "addOnLoad(callback)". Our callback function will be called when all of the code has been loaded into the browser. The following code sequence ensures that the required functions are fully loaded before we try to start the editor:

```
dojo.require ("dijit.Editor");   // loads the code from Google's servers
dojo.addOnLoad(startEditor);    // calls startEditor() after the loading is complete
```

The complete program that allows the user to enter and format text on a web page is downloadable in the following section in a file named "dojotest.html".

You can see a lot of good documentation for Dojo at:
http://docs.dojocampus.org/

dojotest.html

Instructions: This program uses dojo to display an interactive editor on the browser window.

dojotest.html

Midterm Review

The following topics may be on the midterm exam:

- DOM API
- XMLHttpRequest object
- How many logical tasks should one JavaScript function perform?
- parsing XML
  - Given cities.xml, write the method call that would find "Portland"
- using JSON
  - Given cities.json, write the variable name for "Portland"
- server requirements
  - LAMP: Linux, apache, mySQL, php/perl/python/java
- Terms
  - Throughput vs. latency
  - Asynchronous deals with latency
  - framework
  - API
  - asynchronous
  - degradeable

object

JavaScript

loops, branches, functions, arguments, arrays, objects, local vs. global, refactoring

sampleMidterm.html

Instructions: In the attached file, you can find the question in the comment at the top of the file. In the real midterm, you will not be allowed to use a browser to test your work.

sampleMidterm.html

sampleMidterm.js

Instructions: Solve the same problem as you did in "sampleMidterm.html" in the previous section, except this time use function addNode(), which you will find in the attached file. In the real midterm, you won't be able to use a browser to test your solution.

sampleMidterm2.js