Lecture 7: Mashups.
Computer Science E-75
March 31, 2008

Q: If you have PHP, when would you use store procedures?
A: Store Procedures in a typical database allow you to pre-define queries that you run frequently for performance reasons.

Congratulations to our Big Board winners!

Grading for Project 2:
- Similar to project 1. Good really does mean "Good"! Do not assume a straight mapping to a letter grade.

Demo:
    - The News Channel on the Nintendo Wii.
    - There is a globe feature on the News Channel that shows a "pile" of news articles on a specific location. So, for example, Washington D.C. has a large pile of news articles; presumably for politically-related news items.
    - This is the motivation for the next project.

- Mashups
    - A mashup is a combination of two or more existing tools or technologies to create some novel software.
    - In project 3, you will use a combination of the Google Maps API, Google News, and a MySQL database to create a mashup.

Google Maps
    - When it came out, it seemed to blow out of the water the other competitors of the day (e.g. MapQuest)
    - It was the UI enhancements (such as scrolling by simply dragging the map around, without needing to click on direction arrow) that really made it shine.
    - It works via Ajax. When you drag the map around, the page automatically requests the next image to display and seamlessly shows the map.
    - This is nice, but with a live-update page such as this, you cannot simply send the URL to a friend such that they will see the same location.
        - Give it a try: go to maps.google.com, enter a location, and notice how the URL doesn't change from "maps.google.com"
        - Google has gotten around this by providing a "Link to this page" link that contains a link with the state of the page (its zoom level, location)
        - This is an example of how Ajax can break a site and that none of the solutions are ideal, they are all workarounds.
    - So, let's say I really like Google Maps and I want to be able to integrate Google Maps into my own site.
        - Google Maps API: http://code.google.com/apis/maps/
        - This will likely be your friend for the next few weeks!

Quick Review: Latitude and Longitude:
    - Realize that a longitudinal line is a "vertical" line (longitude 0 is in Greenwich, England)
        - Positive longitudes are east of Greenwich.
        - Negative longitudes are west.
    - Latitude is a "horizontal" line (e.g., the equator is at latitude 0)
        - Positive latitudes are north of the equator.
        - Negative latitudes are soutuh.
    There are two forms:
        - degrees, minutes, seconds (e.g., longitude +42º 22' 30", latitude -71º 6' 22")
        - in decimal degrees (e.g., +42.375º, -71.106111º)

- Use http://www.earthtools.org to find latitude and longitude on a Google Maps mashup.

Examples
- Goal: Make a google map that integrates into our webpage.
- Example 1:
  - Notice the script tag: <script src="http://maps.google.com/maps?file=api&v=2&key=ABQIAAAA8igYd929VTmOEMLNjNyP1xSEziMNlTbhQux6Dd1peYjpW-7jNBQ_Jw6QrcRe81uNU5b6imhn7Pc7kg" type="text/javascript"></script>
  - There is a "key" argument.
    - Google asks you to request a key when you use their API. Presumably it maps back to your domain, so you can't use someone else's key.
    - This might be used for tracking purposes, so Google can see who is using their API
  - Now, to load a map into the page you:
    - create a div with some id (in this example, div id="map")
    - insert, into Javascript:

```
if (GBrowserIsCompatible())
 {
     // instantiate map
     var map = new GMap2(document.getElementById("map"));

        // center map on Cambridge
     map.setCenter(new GLatLng(42.375, -71.106111), 15);
 }
```

    - That code is enough to load a Google Map into the example webpage, centered on Cambridge.
    - Notice that, in the above code, we first test if the browser is capable of running Google Maps (with GBrowserIsCompatible() function that comes with the Maps API script)
    - Next, we instantiate a GMap2 class in the map div with "new GMap2(document.getElementByID("map")"
    - The last bit of code centers the map with some latitude and longitude coordinates with a zoom level (where 0 is completely zoomed out, and 16 or 17 is completely zoomed in)
    - You must run the initiate() function when the page is loaded. You can do this by using the "onload" body attribute:

```
<body onload="initialize()" onunload="GUnload()">
```

- Example 2:
  - The first example didn't provide you with a way to change the zoom level! There were no navigation controls.
  - This example fixes that. Notice the new code builds onto the old code with three new calls:
  - map.addControl(new GLargeMapControl()); - runs the addControl method on the maps class. In this case it adds navigation control
  - map.addControl(new GMapTypeControl()); - adds the "Map | Satellite | Hybrid" buttons in the upper right hand corner
  - map.enableScrollWheelZoom() - adds the ability to zoom in to a map with a mouse scroll wheel.
- Example 3:
  - The previous examples weren't really mashups, just less functional versions of Google Maps.
  - It has this new code in the Javascript, after instantiating the map class and setting its center:

```
GEvent.addListener(map, "moveend", function() {
        alert("Where do you think you're going?!");
    });
```

  - Remember from YUI's event handler that there is a way to "listen" for particular events. The same idea exists in Google's APIs, where the page is "listening" for when the map is moved.
  - When the map is moved, it runs the lambda function that runs an alert.
    - Notice that the listener fires on "moveend", which is not specifically moving by

dragging. It is a general move.
        - So, for example, if you use "map.setCenter" after the listener has been added, it will cause the listener function to fire whenever map.setCenter is run.
    - Example 4:
        - Adds the ability to add a marker to your map. The beginning of a mashup!
        - We can add a marker with:
            var marker = new GMarker(new GLatLng(42.3748, -71.1182));
            map.addOverlay(marker);
        - The first line instantiates a GMarker class at a specific set of coordinates.
        - Next, the map.addOverlay method adds that marker to the map itself.
        - This is not very much information - wouldn't it be nice to be able to click the marker and get additional information?
    - Example 5:
        - Provides additional information when you click on a marker on a map.
        - First, we'll define the point that we're interested in:
            var point = new GLatLng(42.3748, -71.1182);
        - Next, we want to set the center of the map at the point (with a zoom level of 17, or the most zoomed in):
            map.setCenter(point, 17);
        - Next, we'll just make a decision to show the hybrid view (satellite + roads) so that someone can see the location of our marker:
            map.setMapType(G_HYBRID_MAP);
        - next we'll create the marker
            var marker = new GMarker(point);
            map.addOverlay(marker);
        - Finally, we'll add the info window to the marker with a listener that listens for a "click" event:

```
            GEvent.addListener(marker, "click", function() {
                var html = "<a
href='http://www.fas.harvard.edu/~ims/Class/harvard202.html'>Harvard Hall 202</a>";
                map.openInfoWindowHtml(point, html);
            });
```

    - Example 6:
        - "Fake Google" implementation. It has an input field that allows us to enter a location and the Maps API will find it.
        - It works by invoking a javascript function when the form was submitted:

```
            function go(address)
            {
                // ensure geocoder exists
                if (!geocoder)
                    return;

                geocoder.getLatLng(address, function(point) {
                    if (!point)
                        alert("Address not found!");
                    else
                    {
                        map.setCenter(point);
                        map.setZoom(15);
                    }
                });
            }
```

        - Notice that it checks for a geocoder (which was an instantiated GClientGeocoder class in the initialize() function)
        - It then invokes the getLatLng method in the GClientGeocoder (since geocoder was assigned such in initialize()) which takes two arguments: the requested address (which is passed from the form) and another lambda function
        - This function accepts a point from Google. If the point is null, that means the address was not found, and the user is alerted as such.
        - Otherwise, the point is not null and was therefore found. We then set the center of

the map to the point and zoom it to level 15.

Google Maps API:
- You can look up all of these functions (and a thorough list of available functions) via the Google Maps API link:
  - http://code.google.com/apis/maps/documentation/reference.html

Shuttleboy.com:
- A web-based re-creation of a UNIX tool David created years ago for undergrads to be able to determine when a shuttle would arrive.
- The program was meant to look up the shuttle schedules and find the next available options
- Enter the world of GPS: now the buses have GPS units in them which relay the coordinates of each shuttle back to a Java applet.
- What seemed like an appealing mashup was to forego the Java applet and instead overlay the live position of each of the shuttles.
- It works by:
  - First, hard-coding the lat & long of shuttle stops into a javascript array.
    - Given that the stops most likely won't change (at least not frequently), hard-coding them seemed an acceptable design decision
  - Instantiate the map and add the controls.
  - To show the stops, it does a typical for loop through each of the points of the shuttle stops array
    - It adds a stylized marker (a blue dot) for each shuttle stop
  - It uses an Ajax call to a PHP file that returns a JSON object of the current locations of the shuttles
  - Every two seconds it contacts the PHP file to obtain a new JSON string.
  - The javascript code parses the string (which contains the coordinates of the available buses) and places a marker for the location of each bus.

Project 3:
- You will need to create a mashup that shows the quantity of news articles for a given area.
- This will be done through an RSS feed (which is just an XML file) which provides a list of news articles for a given zip code.
- Remember, though, that Google maps operates in coordinates. No worries! We have obtained a set of data that maps zip codes to latitude and longitude coordinates.
- You will then need to determine the visible coordinates in a users window, lookup the zip codes that are visible in the window, and then return a reasonable number of news articles for the given zip codes.
- We've given you some MySQL code that will be able to determine the distance in miles between some sets of coordinates.
  - This way, you can figure out the number of miles visible in a google map, and determine how many zip codes are nearby.
- See the project 3 specification for more information.
- A sample implementation plan (in sample version numbers):
  - 0.1: just make a map
  - 0.2: ... + form: give yourself some space for a form in addition to the map
  - 0.3: ... + onsubmit + Gmap2.setCenter: do the onsubmit handler for that form, and call setCenter on the address that was passed in
  - 0.4: ... + GMarker + GInfoWindow: plant a marker in the middle of that zip code, and attach a GInfoWindow that links to some page (perhaps the Google News link for that zip code)
  - 0.5: ... + RSS + Ajax: Use Ajax to get the XML data for the news in the zip code.
  - 0.6: ... + $5.00 table: Now you need to take into account zooming out. So, you need to modify your ajax code to return news articles for zip codes visible in the entire window
  - 0.7: ... + moveend + zoomend: Start listening for moveend or zoomend events and run all of the above code to place new markers or take old ones away.

Final Project:
- Notice that the final project is out!

- The requirements are low, so use your creativity for a project proposal!