CHAPTER 13

Online Maps and 3-D Digital Globes

It would be difficult to overstate the importance of maps in the course of human civilization. Maps help us place ourselves in a spatial context with regard to everything else on the planet. With the advent of the Web, we have been able to access online maps, which have proven to be both useful and fascinating. There are many practical daily uses for these maps: driving directions, locating a restaurant in your neighborhood, thinking of places to go for travel. Online maps let us explore parts of the world in new ways too. Moreover, maps provide an intuitive conceptual and visual metaphor/space for connecting other things; as part of our cultural development, we have all developed a strong intuition for maps.

It is no wonder then that online maps have been used extensively in many mashups. One reason for this extensive activity is that the contemporary online maps are designed for easy customization. In this chapter, you will learn how to do so in this chapter. It is an exciting time for web-based mapping, and we really are only at the beginning of developing this immersive space. Add Global Positioning System (GPS), more immersive systems/platforms like Google Earth and Second Life, ubiquitous computing, GPS devices, and we're going to get amazing stuff.

Note We're good at reading maps and we know how things on maps are related to each other. We're used to adding dots, drawing lines. Hence, it's not much of a stretch for us to add other things – dots, lines, pictures, even more abstract data) to maps. Things that are located in space have a natural spot on maps. That's what I mean by saying that maps are a powerful metaphor.

The goal of this chapter is to provide an introduction into how to use some leading systems for remix purposes (Google maps, Yahoo maps, Microsoft maps, MapQuest, and Google Earth), looking for commonalities and differences. A potential framing question, technically, is how to write a wrapper so that one can substitute one system for another. However, for most people, they just want to use one of these maps and do easy customization – hence I will need to show how you will be able to do so. Each system has strengths and it would be useful to be able to interchange information among them without much effort. Obviously, we will not attempt to exhaust this very rich subject, but provide you with a strong starting point to build on.

In this chapter, I will:

* describe how to use the APIs of the major map providers, such as Google maps, Yahoo maps, virtual Earth, and MapQuest.

* describe how you can make web-based maps without programming
* describe declarative approaches to working with maps, such as creating KM and geoRSS and CSV.

To learn more about the subject on online maps, please read:

* *Building Google Maps Applications with PHP* (Apress)
* *Building Google Maps Applications with Rails* (Apress)

The rise of online maps

The capability of individual users to make web-based digital maps has been rapidly increasing over the last several years. Online maps have evolved quickly from maps with only predefined purposes (e.g., driving directions) to increasingly customizable platforms. That is, we are close to giving maps for the masses, GIS (Geographic Information System) for dummies – maps, maps, maps, everywhere.

Perhaps the most dramatic revelation of the capabilities of what would later be known as AJAX was the emergence of Google Maps in February 2005.[1] It was a watershed event for all web apps. It showed that it was possible to have highly interactive apps but on a big scale. (Yes, people had been using Javascript for menus but not for shipping a substantial amount of realtime data.) In the area of maps, it marked the beginning of what I will refer to as “new-style” online maps as opposed to “old-style” online maps (which are being an endangered species, it would seem. Even mapquest.com has switched over to AJAX type maps.) By "old-style" online maps, I mean non-JavaScript powered maps, onese in which moving around or zooming means reloading the page.

The most obvious aspect of the new style maps is the substantial increase in interactivity (with the fluid drag and drop capabilities, instead of click and wait for a page reload). However, hackers quickly realized that the AJAX technology also allowed Google Maps to be extended to new purposes.[2] Apps that showed up included housingmaps.com. These apps, however, involved extensive reverse engineering of Google Maps; techniques that emerged could break anytime and the whole enterprise was of questionable legality and longevity. What Google did was then smart and novel: it released an API to formalize and regulate the usage of its maps, transforming Google Maps mashups into legitimate business. Google really did transform the whole endeavor of GIS through Google Maps by making online maps accessible to and customizable by the masses. Competitors soon followed. Yahoo, Microsoft, and eventually Mapquest all went new-style, with the release of not only AJAX implementations but APIs to boot.

Examples of Map-based mashups

Before we figure out how to make map-based mashups, it would be handy to look at a number of examples to understand what is possible. As discussed in Chapter 1, here are so

[1] http://en.wikipedia.org/wiki/Google_maps and
http://www.adaptivepath.com/publications/essays/archives/000385.php

[2] http://www.oreillynet.com/etel/blog/2005/05/hackers_tap_into_the_functiona.html

many map-based mashups, including housingmaps.com, chicagocrime.org, and the 1000+ map-based mashups listed on programmableweb.com (http://www.programmableweb.com/tag/mapping) Here are some specific examples to look at:

* http://tutorlinker.com/ -- which connects tutors to students via a mapping interface
* flashearth.com is a mashup of various major online map services displayed through a Flash interface.

Programmableweb shows Google Maps as by far the popular API used in mashups[3]. Mapping as a category is very popular: Yahoo Maps and Microsoft Virtual Earth are also in the top ten.

Declarative vs Procedural Approaches

I've partitioned creating maps into two regimes: making maps without programming and a programmatic approach to maps, which roughly corresponds to declarative vs procedural approaches.

Making maps without programming: mapbuilder.net

Google Maps now has "My Maps". With the Microsoft Virtual Earth interface, it is possible to assemble your own collection. I need to update this section to reflect advances in these products.

The scenario I want to focus on in this chapter is building a map with dots pointing to a list of places for which you have addresses in the United States. We will look at more sophisticated scenarios later.

Let's see what how much of a Google Map you can build without any programming. Let's use mapbuilder.net. (There seems to be other similar services such as blipstar.com, which seems to require a payment.)

What are some sites built using mapbuilder.net? Drawing from the list of featured maps[4], I see ones like Edinburgh Pub Guide[5]. Note some have been heavily used in commercial contexts: the most popular of all time for mapbuilder.net-built maps[6] is the " Find a Distributor " map for Pacific Wireless.[7]

---

[3] http://programmableweb.com/apilist/bymashups

[4] http://www.mapbuilder.net/About.php

[5] http://www.imkblue.pwp.blueyow 2bMuepsta.uw.imk/-5(i(er.nM)6(u)-4(x5(wp))7([(h)mlvily)php)]TJEMC /P /MC2D 21 BDC

Mapbuilder.net How-to

1. Sign up for an account.[8]

2. Select the "New Map" link.[9] (Note that "Map Name must only contain letters, numbers, underscores, and dashes.")

3. You can create a dot in one of two ways:

   * Start typing in addresses to add. (under "Location Search & Quick Navigation").

   * Click on the map to indicate the spot's location.

4. When you add a new dot, it will be flashing. Click on the any marker on the map to update its information or delete it (use 'Update' and 'Delete' buttons).

5. Use "Save Center, Zoom, MapType" button located on the map to save center of your map, zoom level and map type (regular map, Satellite, Hybrid) as well. (You will have to adjust the scale and center of the map (by using the zoom control dragging the mouse) to fit your taste; the default I find is not helpful.)

6. Use the preview button to take a look at the map.

With a bit of futzing, and by doing a series of searches on Yahoo local for addresses, I created "Some of my favorite bookstores around Berkeley"[10], as hosted on mapbuilder.net. Mapbuilder.net also permit you host the map elsewhere, embedding either Yahoo maps or Google Maps.:

* "Some bookstores I like" (Google version)[11], kept up-to-date via javascript injection.[12]

* "Some bookstores I like" (Yahoo version)[13]

Overall, I would recommend mapbuilder.net to you as a way to quickly build custom Google or Yahoo maps or as a way to get start learning the APIs.[14]

I want to also test the import feature for mapbuilder.net, something one would expect to have. (e.g., I have a list of places to visit and want to see them all on a map. Restaurants in a neighborhood, people to organize carpools for.)

[8] http://www.mapbuilder.net/SignUp.php

[9] http://www.mapbuilder.net/Map.Add.php

[10] http://www.mapbuilder.net/users/rdhyee/9329

[11] http://raymondyee.net/maps/SomeBookstoresGMap.html

[12] For an explanation of how to use javascript injection: http://www.mapbuilder.net/Map.Implemenation.php

[13] http://raymondyee.net/maps/SomeBookstoresYMap.html

[14] One thing that might be a bug though: I wasn't able to delete a certain point using Firefox 1.5.0.7 on XP no matter what I did. I finally deleted the point by logging in to mapbuilder.net using Opera. Any idea what was going on? Others have problems deleting points? I reported the problem but got no response.

A mashup opportunity: Mapping Yahoo local collections

Yahoo Local makes it easy to make collections of places, but it does not allow easy mapping of those places. For instance, I assembled a collection of bookstores around Berkeley,[15] but the Yahoo interface does not allow me to easily map those bookstores. In this section, we will create is by transforming one data format to another (specifically, the address is a flea bookstores offered up in the XML that comes from the Yahoo local API in to CSV that is understood by mapbuilder.net. There are two steps to this, which we will cover below:

1. Get information out via the Yahoo Local API
2. Transform the collection appropriately (XML to CSV) to feed into mapbuilder.net

Getting XML out of Yahoo Local via getCollection

We will use the getCollection method of the Yahoo! Local Web Services, which "enables you to get detailed information about a collection created with Yahoo! Local Collections, through a REST-like API".[16] How to use it?

The base URL is:
`http://collections.local.yahooapis.com/LocalSearchService/V1/getCollection`

There are four parameters

query name	function	the value for our query
appid	the application ID.	raymondyee.net
collection_id	the ID of the collection to be retrieved.	1000014156
output	the output type	*unspecified*, thus returning the default of XML
callback	specifies a callback function	*unspecified*

In other words, we formulate the following query
`http://collections.local.yahooapis.com/LocalSearchService/V1/getCollection?appid=raymondyee.net&collection_id=1000014156`
that returns:

Listing 13-??? XML for a collection from Yahoo! Local

```
<?xml version="1.0" encoding="UTF-8" ?>
<Result id="1000014156" xmlns="urn:yahoo:travel"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="unknown">
  <Title>bookstores around Berkeley</Title>
  <Description>some of my favorite bookstores around Berkeley.</Description>
  <CreatedTime>2006-10-24 13:29:40</CreatedTime>
  <Username>Raymond Yee</Username>
  <CommentCount>0</CommentCount>
  <Item>
    <Address>
```

[15] http://local.yahoo.com/collections?cid=1000014156

[16] http://developer.yahoo.com/local/V1/getCollection.html

```
      <Address1>2476 Telegraph Ave</Address1>
      <Address2 />
      <City>Berkeley</City>
      <State>CA</State>
      <PostalCode>94704</PostalCode>
    </Address>
    <id>21518795</id>
    <Title>Moe&apos;s Books</Title>
    <CreatedTime>2006-10-24 13:29:41</CreatedTime>
    <Description>* The Bay Area&apos;s Largest Selection of Used Scholarly
Books</Description>
<Url>http://local.yahoo.com/details?id=21518795&amp;stx=&amp;csz=Berkeley+CA&amp;ed=
xoDOxa160SyYoswS6OvDhQk64pj4Q8RHG5PQhcSqprzxVT6mDHMezwfQ2U244pugG4LDSdibA78iSw--
</Url>
    <type>Retail Shopping</type>
    <Category>Used &amp; Rare Bookstores</Category>
    <Photo />
    <Tag />
    <Phone>(510) 849-2087</Phone>
  </Item>
  <Item>
    <Address>
      <Address1>1730 4th St</Address1>
      <Address2 />
      <City>Berkeley</City>
      <State>CA</State>
      <PostalCode>94710</PostalCode>
    </Address>
    <id>21512172</id>
    <Title>Cody&apos;s Books</Title>
    <CreatedTime>2006-10-24 14:07:28</CreatedTime>
    <Description />
<Url>http://local.yahoo.com/details?id=21512172&amp;stx=&amp;csz=Berkeley+CA&amp;ed=
3uqWba160SzFEqntYzu46yunejqBEmJnCBEi_I7QbD68sZTEVRYl4WkOGEf6alVIaEB3</Url>
    <type>Retail Shopping</type>
    <Category>Bookstores</Category>
    <Photo />
    <Tag />
    <Phone>(510) 559-9500</Phone>
  </Item>
  <Item>
    <Address>
      <Address1>6060 El Cerrito Plz</Address1>
      <Address2 />
      <City>El Cerrito</City>
      <State>CA</State>
      <PostalCode>94530</PostalCode>
    </Address>
    <id>21414999</id>
    <Title>Barnes &amp; Noble Booksellers</Title>
    <CreatedTime>2006-10-24 14:07:56</CreatedTime>
    <Description />
```

```
<Url>http://local.yahoo.com/details?id=21414999&amp;stx=&amp;csz=El+Cerrito+CA&amp;e
d=Fo51gq16OSy24fPx_u7IvyZen3kxQq5wR9ZOi_Aos2J.pPlJ75D_th3K2MHtNCWF_V5k_nOq62ssy3I-
</Url>
    <type>Retail Shopping</type>
    <Category>Bookstores</Category>
    <Photo />
    <Tag />
    <Phone>(510) 524-0087</Phone>
  </Item>
</Result><!-- fca2.travel.scd.yahoo.com uncompressed/chunked Fri Dec  1 11:24:33 PST
2006 -->
<!-- ws01.search.scd.yahoo.com uncompressed/chunked Fri Dec  1 11:24:33 PST 2006 -->
```

Tranforming the Yahoo! Local XML into CSV for mapbuilder.net

The goal is to convert this XML data into CSV. There are various techniques you could consider to do this. One way is to write XSLT to transform the Yahoo XML to CSV :

```
   <?xml version="1.0" encoding="UTF-8"?>
   <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:yahoo="urn:yahoo:travel">
     <xsl:output method="text" version="1.0" encoding="ISO-8859-1" media-
type="text/plain"/>
     <!-- root -->
     <xsl:template match="yahoo:Result">"Caption","Street Address
","City","State","Zip"
       <xsl:apply-templates select="yahoo:Item" mode="MapToRows"/>
     </xsl:template>
     <!--Item -->
     <xsl:template match="yahoo:Item" mode="MapToRows">"<xsl:value-of
select=".//yahoo:Title" disable-output-escaping="yes"/>","<xsl:value-of
select=".//yahoo:Address//yahoo:Address1"/>","<xsl:value-of
select=".//yahoo:Address//yahoo:City"/>", "<xsl:value-of
select=".//yahoo:Address//yahoo:State"/>", "<xsl:value-of
select=".//yahoo:Address//yahoo:PostalCode"/>"
       </xsl:template>
   </xsl:stylesheet>
```

Note: There is one problem with the XSLT that I've not yet been able to fix -- that of escaped XML showing up in the output. I should come back to [http://www.dpawson.co.uk/xsl/sect2/N2215.html#d3496e394 Disable Output Escaping]).

Another way is to write a PHP script that takes a collection id and outputs CSV:[17]

```
<?php
```

[17] http://examples.mashupguide.net/ch13/yahooCollectionToCSV.php

```
function getResource($url){
  $chandle = curl_init();
  curl_setopt($chandle, CURLOPT_URL, $url);
  curl_setopt($chandle, CURLOPT_RETURNTRANSFER, 1);
  $result = curl_exec($chandle);
  curl_close($chandle);
  return $result;
}

// get a collection_id
  $cid  = isset($_REQUEST['cid']) ? $_REQUEST['cid'] : "1000014156";  //default to
my own

  $url =
"http://collections.local.yahooapis.com/LocalSearchService/V1/getCollection?appid=ra
ymondyee.net&collection_id=". urlencode($cid);
  $feed = getResource($url);
  $xml = simplexml_load_string($feed);


  //header("Content-Type:text/csv");
  $out = fopen('php://output', 'w');

  $header = array("Caption","Street Address","City","State","Zip");
  fputcsv($out, $header);

   foreach ($xml->Item as $item) {
    $caption = $item->Title;
    $street_address = $item->Address->Address1;
    $city = $item->Address->City;
    $state = $item->Address->State;
    $zip = $item->Address->PostalCode;
    fputcsv($out, array($caption,$street_address,$city,$state,$zip));
  }

    fclose($out);
?>
```

With this code in hand, we can generate a CSV file that we can feed to mapbuilder.net:

`http://examples.mashupguide.net/ch13/yahooCollectionToCSV.php?cid=1000014156`

To try this out, I can go to my collections

`http://local.yahoo.com/userreviews?target=pOTJ1rUjf64lpQPwpZGZmXVTOyaM&rvwtype=COLLECTION`

and pull out collection id numbers to feed to the script to generate the CSV:

```
Caption,"Street Address",City,State,Zip
"Moe's Books","2476 Telegraph Ave",Berkeley,CA,94704
"Cody's Books","1730 4th St",Berkeley,CA,94710
"Barnes & Noble Booksellers","6060 El Cerrito Plz","El Cerrito",CA,94530
```

You can then take the CSV and feed it to mapbuilder.net to create your map.

Note: You might say that writing such a script only gets you CSV and you still have to manually create a map with mapbuilder.net – and you're right. Later, we'll transform this collection into formats that are easier to work with in terms of generating maps.

Collection building in Google Maps and Virtual Earth

With Microsoft virtual Earth, the story is a bit more complicated:

* local.live.com has excellent collection building facilities. I was surprised how easy it was to look up bookstores in and save them to collections and then to see those bookstores on the map all within the Microsoft map environment. If you make a collection public, you can share it. [18]
* the VE widget can create a layer that is built from public Live Maps collections.[19] I have been hoping for the equivalent to Yahoo's API to get an XML representation of a collection, but I have not found a way to get the XML representation of a collection.[20] At best right now, in the `addLayer` method of the VE widget, you can specify a callback function which is fed a collection of pushpins that represents the layer. I suppose that one can rig up a combination of a webpage that reads in collections and a server-side script that receives that information as a rough equivalent of the Yahoo API.[21] There is always the option of reverse-engineering the VE widget -- but that's rather painful.

Bottom line: mapbuilder.net is not that easy to use to place a lot of dots if you don't already have the addresses nicely laid out in a spreadsheet already. It would be nice to use a service like Yahoo local to pull up addresses and then pass the info into mapbuilder.net – but that would defeat the ease of use advantage of mapbuilder.net. live.local.com is surprising easy to use for building maps – but it's not easy to extract address information to create maps on competing map services.

A new wrinkle. If you want to build a Google or Yahoo map easily, mapbuilder.net is one of the best solutions. If you want to build a Virtual Earth map, I'd just use live.local.com, saving the map as a public live collection, and then embedding it elsewhere using a bit of javascript code.

---

[18] http://maps.live.com/?v=2&cid=74B8FFD299EDD840!106

[19] http://dev.live.com/virtualearth/sdk/Ref/HTML/WorkingWithLayers.htm

[20] e.g., http://collections.local.yahooapis.com/LocalSearchService/V1/getCollection?appid=raymondyee.net&collection_id=1000014156 to get an XML representation of http://local.yahoo.com/collections?cid=1000014156

[21] More documentation on the VE widget: http://virtualearth.spaces.live.com/blog/cns!2BBC66E99FDCDB98!5265.lists new features people would like. http://dev.live.com/virtualearth/sdk/Ref/HTML/WorkingWithLayers.htm and http://msdn2.microsoft.com/en-us/library/aa907682.aspx are further places to look.

Data Exchange Formats

CSV

Comma separated values. It's not sexy or cool but what you are going to get out of a spreadsheet and what we can feed into mapbuilder.net. Here's what mapbuilder.net has to say about CSV:[22]

> *To import and geocode locations it's necessary to upload CSV (The **Comma** Separated Value) file. CSV files can be easily generated from any data source e.g. use "Save As", then select "Save as Type" - "CSV (comma delimited)" to save your Excel work book as CSV file.*
>
> *After uploading file to the MapBuilder it will appear in "Uploaded Files List" below and you'll able to geocode addresses from that file and import locations to active map.*
>
> *Basically for a successful geocoding it's necessary to have a **full addrees** column (e.g. "123 Main Street, Bellwood, PA") or separate columns for **Street Address** ("123 Main Street"), **City** ("Bellwood"), **Zip** ("16617") and **State** ("PA"). State, Zip columns are optional but they will help to receive correct geocoding results.*
>
> *For a successful import to the MapBuilder system it's necessary to have at least "Caption" column in your CSV file.*
>
> *After geocoding you'll see locations plotted on small maps and once you will ensure that everything is correct you'll able to import locations to your map.*

One major caveat about discussing CSV is that CSV is really not a standard or default format for enterprise systems. Yet, CSV persists.

GeoRSS

Why geoRSS is interesting and some background? GeoRSS is supported in Yahoo's simple V1[23] It seems that geoRSS[24] has the air of being a standard. According to Mikel Maron:[25]

> *GeoRSS is the easiest and most effective way to share and build maps. GeoRSS is supported by Yahoo!, worldKit, and many others, and is on its way to*

[22] http://www.mapbuilder.net/import/

[23] http://developer.yahoo.com/maps/simple/index.html

[24] http://www.georss.org/w3c.html and http://esw.w3.org/topic/GeoInfo

[25] http://brainoff.com/gmaps/mgeorss.html

> *standardization. Wouldn't it be great if Google joined up and supported a common way to communicate place on the web? How 'bout it Google?*

GeoRSS is a way of embedding location information within other formats, such as RDF and XML (e.g., RSS). If you go to what seems to be an official site[26], you will find a representative example of a geoRSS expression (where the georss namespace refers to `http://www.georss.org/georss`:

```
<georss:where>
  <gml:Point>
    <gml:pos>45.256 -71.92</gml:pos>
  </gml:Point>
</georss:where>
```

and, in a simpler variation:

```
<georss:point>45.256 -71.92</georss:point>
```

Note the contrast to the more ubiquitous way of embedding geo-information, which seems to also go under the name GeoRSS:

```
<geo:lat>55.701</geo:lat>
<geo:long>12.552</geo:long>
```

How to explain this discrepancy? The latter is technically formalized in the W3C Basic Geo Vocabulary[27] -- but it is recommended that it still be recognized under GeoRSS:[28]

> *However, W3C Geo has great inertia and will continue to be an important format. It is recommended that GeoRSS Parsers support W3C Geo. In time, the hope is that these two namespaces can be reconciled.*

Things are a bit complex since many parties are using the georss moniker but they all mean something different -- as well see when we talk about Yahoo's use of geoRSS.

Yahoo's use of GeoRSS

The Yahoo simple API[29] and v2[30] allows one to pass in Yahoo's take on GeoRSS, which is RSS 2.0 with the following extensions:

* `geo:lat` and `geo:long` elements in both the channel and item RSS elements, where the geo namespace is `http://www.w3.org/2003/01/geo/wgs84_pos#`
* elements in the `ymaps` (`http://api.maps.yahoo.com/Maps/V1/AnnotatedMaps.xsd`) namespace, including `ymaps:Address`, `ymaps: CityState`, `ymaps: Zip`, and `ymaps:Country` to denote an address associated with an `item`.

[26] http://www.georss.org/

[27] http://www.w3.org/2003/01/geo/

[28] http://www.georss.org/w3c.html

[29] http://developer.yahoo.com/maps/simple/V1/reference.html

[30] http://developer.yahoo.com/maps/georss/index.html

This ability to use either `geo:` and `ymaps:` extensions reflects the ability of Yahoo maps to do the geocoding for you: you don't need the lat/long info if you have an address. (We will look later at how to call on services to do explicit geocoding.)

Let's look at a working example. I converted the Cafe Yahoo local XML to geoRSS to feed to the simple API of Yahoo maps:[31]

```
    <?xml version="1.0" encoding="UTF-8"?>
    <rss xmlns:geo="http://www.w3.org/2003/01/geo/wgs84_pos#"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:yahoo="urn:yahoo:travel"
xmlns:ymaps="http://api.maps.yahoo.com/Maps/V1/AnnotatedMaps.xsd" version="2.0">
      <channel>
        <title>Nice cafes around Berkeley</title>
        <link>http://raymondyee.net/maps/</link>
        <description/>
        <item>
          <title>Yali's Oxford Street Cafe</title>
          <link>http://raymondyee.net/</link>
          <description>Yali's Oxford Street Cafe</description>
          <ymaps:Address>1920 Oxford St</ymaps:Address>
          <ymaps:CityState>Berkeley, CA</ymaps:CityState>
          <ymaps:Zip>94704</ymaps:Zip>
          <ymaps:Country>US</ymaps:Country>
        </item>
        <item>
          <title>Le Bateau Ivre</title>
          <link>http://raymondyee.net/</link>
          <description>Le Bateau Ivre</description>
          <ymaps:Address>2629 Telegraph Ave</ymaps:Address>
          <ymaps:CityState>Berkeley, CA</ymaps:CityState>
          <ymaps:Zip>94704</ymaps:Zip>
          <ymaps:Country>US</ymaps:Country>
        </item>
        <item>
          <title>Village Grounds</title>
          <link>http://raymondyee.net/</link>
          <description>Village Grounds</description>
          <ymaps:Address>1797 Shattuck Ave</ymaps:Address>
          <ymaps:CityState>Berkeley, CA</ymaps:CityState>
          <ymaps:Zip>94709</ymaps:Zip>
          <ymaps:Country>US</ymaps:Country>
        </item>
        <item>
          <title>Musical Offering Classical CD'S Cafe Bistro</title>
          <link>http://raymondyee.net/</link>
          <description>Musical Offering Classical CD'S Cafe Bistro</description>
          <ymaps:Address>2430 Bancroft Way</ymaps:Address>
```

[31] http://examples.mashupguide.net/ch13/cafes.georss.xml

```
      <ymaps:CityState>Berkeley, CA</ymaps:CityState>
      <ymaps:Zip>94704</ymaps:Zip>
      <ymaps:Country>US</ymaps:Country>
    </item>
    <item>
      <title>Cafe Milano</title>
      <link>http://raymondyee.net/</link>
      <description>Cafe Milano</description>
      <ymaps:Address>2522 Bancroft Way</ymaps:Address>
      <ymaps:CityState>Berkeley, CA</ymaps:CityState>
      <ymaps:Zip>94704</ymaps:Zip>
      <ymaps:Country>US</ymaps:Country>
    </item>
  </channel>
</rss>
```

Go to

```
http://api.maps.yahoo.com/Maps/V1/annotatedMaps?appid=raymondyee.net&xmlsrc=
http://examples.mashupguide.net/ch13/cafes.georss.xml
```

and you will see the rendition of the location data in V1 of the Yahoo maps.

You can also pass the geocoded RSS 2.0 the AJAX Yahoo maps.[32] One way to see this functionality at work is to:

1. Copy code from one of the Yahoo examples[33] and adapt it to be centered around the UC Berkeley campus.[34]

2. Bring up that example. Invoke the Javascript shell on it[35] and type `map.addOverlay(new YGeoRSS('http://examples.mashupguide.net/ch13/cafes.georss.xml'));` to load the geoRSS file into the map.

USING THE DOM INSPECTOR AND JAVASCRIPT SHELL TO LEARN THE API OF AJAX APPS

In learning these mapping APIs (as well as other of Ajax applications), I recommend using Firefox, the DOM Inspector[36] and the JavaScript Shell (bookmarklet)[37] to manipulate Google maps "live". The combination allows for "live" interaction with the little apps; you can load a running map, analyze the details of how it is working while running, and issue commands that take immediate effect.

[32] http://developer.yahoo.com/maps/georss/index.html

[33] http://developer.yahoo.com/maps/ajax/V3/ajaxexample1.html

[34] http://examples.mashupguide.net/ch13/yahoo.map.1.html

[35] http://www.squarefree.com/shell/

[36] http://www.codestore.net/store.nsf/unid/BLOG-20050228 and http://kb.mozillazine.org/DOM_Inspector

[37] http://www.squarefree.com/shell/

The DOM Inspector allows you to look at the HTML DOM as a tree and make changes in that DOM. The DOM Inspector comes with Firefox, but not with the default installation options on Windows.

The Javascript shell is a great way to learn the inner workings of an AJAX widget.

A good write-up about how to use the javascript shell is to be found in Mark Pilgrim's *Greasemonkey Hacks,* Hack 10, Section 1.11.4[38]: "JavaScript Shell is a bookmarklet that allows you to evaluate arbitrary JavaScript expressions in the context of the current page. You install it simply by dragging it to your links toolbar. Then you can visit a web page you want to work on, and click the JavaScript Shell bookmarklet in your toolbar. The JavaScript Shell window will open in the background."

You can use the Javascript shell to test out little code segments. Try it out -- and if it works, then add it to your Javascript code.

I talk about how to install these tools in Chapter 8. Should I just refer readers to that place?

End Sidebar

Problems with the Yahoo simple API

I have two particular problems with using V1 or the Yahoo Simple Map API:

1. Uninformative error messages. In creating XML to pass into the API, in which I did not include the item elements inside the channel tag, (I fortunately used feedvalidator.org to help me out....) I got the unspecific error message: http://api.maps.yahoo.com/maps?syne=YAHOO_MAPS_MISSING_ITEMS_TAG I've also gotten: http://api.maps.yahoo.com/maps?syne=-8010&xe=2&xl=1&xc=49 -- I think when the XML file doesn't exist. *Why kind of error message is that????*

2. Timeout problems: I reported this problem[39], got a response[40] -- but I did run into a timeout problem recently.

GeoRSS in Virtual Earth

VirtualEarth also has the capacity to render geoRSS files, specifically the following formats (directly quoting the article)[41]

1. The geo:lat and geo:long elements.

```
<geo:lat>42.5</geo:lat>
<geo:long>12.5</geo:long>
```

2. The traditional geo:Point format, which uses geo:lat and geo:long elements.

```
<geo:Point>
   <geo:lat>42.5</geo:lat>
   <geo:long>12.5</geo:long>
```

[38] http://search.safaribooksonline.com/0596101651/greasemonkeyhks-CHP-1-SECT-11

[39] http://tech.groups.yahoo.com/group/yws-maps/message/102

[40] http://tech.groups.yahoo.com/group/yws-maps/message/131

[41] http://dev.live.com/virtualearth/sdk/Ref/HTML/WorkingWithLayers.htm

```
</geo:Point>
```

3. The GeoRSS Simple point format.

```
<georss:point>42.5 12.5</georss:point>
```

4. The GeoRSS GML point format.

```
<gml:Point>
   <gml:pos>42.5 12.5</gml:pos >
</gml:Point>
```

Now I should be able to feed that file to VirtualEarth. Making a slight modification to some sample code[42] to read a geocoded version of the Berkeley cafe example above[43] to come up with the following demo code:

```
<html>
  <head>
  <title>Demonstration of GeoRSS in Virtual Earth</title>
  <!-- based on http://brainoff.com/test/ve.html -->
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <script src="http://dev.virtualearth.net/mapcontrol/v3/mapcontrol.js"></script>
    <script>
      var map = null;
      var layerid=1;
      function GetMap() {
        map = new VEMap('myMap');
        map.LoadMap();
      }
      function AddMyLayer(source){
        var veLayerSpec = new VELayerSpecification();
        veLayerSpec.Type = VELayerType.GeoRSS;
        veLayerSpec.ID = layerid;
        veLayerSpec.LayerSource = source;
        veLayerSpec.Method = 'get';
        veLayerSpec.FnCallback = onFeedLoad;
        map.AddLayer(veLayerSpec);
        layerid++;
      }
      function onFeedLoad(feed)  {
        alert('RSS or Collection loaded. There are '+feed.length+' items in this 
list.');
      }
    </script>
  </head>
  <body onload="GetMap(); AddMyLayer('cafes.georss.2.xml');">
    <div id='myMap' style="position:relative; width:400px; height:400px;"></div>
    Loading <a href="cafes.georss.xml">cafes.georss.2.xml</a>
  </body>
</html>
```

[42] http://brainoff.com/test/ve.html VE demo of georSS

[43] http://raymondyee.net/maps/cafes.georss.2.xml

Note that this example works in Internet Explorer but not in Firefox.

KML

KML, or Keyhole Markup Language, "is an XML grammar and file format for modeling and storing geographic features such as points, lines, images, and polygons for display in Google Earth, Google Maps, and Google Maps for mobile."[44] Google's backing of KML makes it an very important format for the exchange of geographic information.

KML has moved beyond its use in Google Earth alone. For instance, you can display KML files and export search results and one of your "my maps" from Google Maps in KML. Other applications are beginning to support KML: For instance, you can get KML coming out of Yahoo! Pipes.[45] There is support for KML in Feed Validator.[46] KML is being shepherded through a standards process.[47] Google is advising people to use KML so that its geo--search can index KML -- in KML 2.2, there is an attribution element. Google apparently will also index GeoRSS.

See more on KML in the Google Earth section.

Interoperability among formats: geoRSS, KML

In commenting on the Where 2.0 conference, Benjamin Christen wrote: "There are two important XML schemas covered today at Where 2.0 -- GeoRSS and KML."[48] Can we get these two formats -- and others of importance -- to work together?[49] Mikal Maron provides an intriguing portrait of the relationship among various formats:[50]

> *There are of course other geodata formats in use, which deserve a look as alternatives to GeoRSS. KML is used in Google Earth, and loads of data layers have been published by an active community. However, KML is very tied to its application, with features specifically aimed for 3D spinny globes, and the spec is controlled by a single organization. GPX, for data interchange between GPS units, is again very tied to specifics of GPS units. GML is a feature rich vocabulary for encoding geographic information, but its complexity has been daunting for unversed developers, and its proper use misunderstood. GML is similar to RDF, defining a number of primitive objects that can be assembled*

44 http://earth.google.com/kml/

45 http://blog.pipes.yahoo.com/2007/05/02/pipes-adds-interactive-yahoo-maps-kml-support-and-more/

46 http://googleearthuser.blogspot.com/2007/05/feed-validator.html

47 http://geotips.blogspot.com/2007/04/kml-ogc.html

48 http://www.oreillynet.com/conferences/blog/2006/06/georss_and_kml.html

49 http://www.ogleearth.com/2006/05/georss_is_here.html is a good summary of some of the issues.

50 http://xtech06.usefulinc.com/schedule/paper/56

into Profiles for particular purposes. In fact, a GML profile for GeoRSS is a result of the new standard.

Not surprising, Mikal Maron is involved such interoperability efforts as:

* MGeoRSS, an extension that "integrates basic GeoRSS support directly into Google Maps."[51]
* OpenStreetMap, "a project aimed squarely at creating and providing free geographic data such as street maps to anyone who wants them."[52]

Creating Maps with programming

In this section, I will discuss the APIs of various popular online services, specifically, Google, Yahoo, Microsoft, MapQuest. In the previous section, I discussed data formats with an eye to making maps without any (or much) programming. In this section, we would do a bit of programming.

Let me summarize what these maps can do in general. The four covered here all allow you to:

1. embed an AJAX based map, to which you can add custom locations with pop-up windows
2. geocode addresses (translate an address to latitude and longitude), at least for US and Canadian addresses
3. show the maps at various zoom levels and of various types (road, aerial, or hybrid)
4. let you add lines to the maps, to represent things like driving directions.

I would like to flesh out this previous list. I should also explicitly compare the various maps in their specific simiarlities and differences through a table and include info on: location of documentation, location of discussion forums, how the key is used, usage limitations (# calls), whether you can use the maps commercially. Maybe then a second table with the various javascript snippets to do various things (zoom in; adding an overlay, geocoding) -- at the end of the section.

What's the overall strategy in this section? With a bit of copying and pasting, you can get working examples of each of the maps. You can then modify them incrementally. Using the DOM Inspector and the JavaScript shell, you can even make changes to live working examples within the browser. I will also use those mechanisms to highlight important capabilities and functions of the maps.

[51] http://brainoff.com/gmaps/mgeorss.html

[52] http://wiki.openstreetmap.org/index.php/Main_Page

Google Maps API

Now we look at the Google Maps API. We will start with how to embed a Google map using the Google Maps API. The online documentation on how to get started with the maps at the Google website are good. [53] The approach given there, one that I can certainly recommend, is to give you source code for increasingly more complex examples, that you either copy and paste to your own site. I can recommend that approach.

What I offer here is setting up a simple map and then using the Javascript shell to let you work with a "live map" so that you can invoke a command and see an immediate response. The intended effect is that you see the widgets as dynamic programs that respond to commands, whether that command comes in a program or from you entering that command one by one.

Getting started with Google Maps and Javascript Shell

We will use the Google Maps API to make a simple map.

1. Make sure you have a public web directory to host your map and know the URL of that directory. Any Google Map that uses the free, public API, needs to be publicly visible

2. Go to the signup page for a key to access the Google Maps.[54] You will need a key for any given domain in which you host Google maps. (It is through these keys that Google regulates the use of the Google maps API.)

3. Read the terms of service[55] and if you agree to it, enter the URL directory on the host that you want to place your test file. For example, in my case, the URL is `http://examples.mashupguide.net/ch13/`. When I type in this URL, I get the following key:

```
ABQIAAAAdjiS7YH6Pzk2Nrli02b5xxR10RG5t-
vK3TwPKbpNUO2c5sYb4RTmySs_TEFzYvlZrCaYJKlmTzJ5lA
```

   Since your domain will be different, your key will be different. Take down that key. (You can, in fact, enter `http://examples.mashupguide.net/ch13/` and you should get the same key as I list here to confirm that you are doing the right thing.)

4. Copy and paste the HTML code into your own page on your web hosting directory. You should get something like my own example:[56]

[53] http://www.google.com/apis/maps/documentation/#Introduction

[54] http://www.google.com/apis/maps/signup.html

[55] http://www.google.com/apis/maps/terms.html

[56] http://www.google.com/maps/api_signup?url=http%3A%2F%2Fexamples.mashupguide.net%2Fch13%2F

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="content-type" content="text/html; charset=utf-8"/>
    <title>Google Maps JavaScript API Example</title>
    <script
src="http://maps.google.com/maps?file=api&amp;v=2&amp;key=ABQIAAAAdjiS7YH6Pzk2NrliO2
b5xxTO2tB7zlPmV4JaKyl2exQNZXIJ_xTOHgLDumjgyGWFarOlCOJGtI6scw"
      type="text/javascript"></script>
    <script type="text/javascript">

    //<![CDATA[

    function load() {
      if (GBrowserIsCompatible()) {
        var map = new GMap2(document.getElementById("map"));
        map.setCenter(new GLatLng(37.4419, -122.1419), 13);
      }
    }

    //]]>
    </script>
  </head>
  <body onload="load()" onunload="GUnload()">
    <div id="map" style="width: 500px; height: 300px"></div>
  </body>
</html>
```

5. Now make one modification to the example by adding the line

```
window.map = map
```

after

```
var map = new GMap2(document.getElementById("map"));
```

to expose the map object to the Javascript shell utility[57]:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="content-type" content="text/html; charset=utf-8"/>
    <title>Google Maps JavaScript API Example</title>
    <script
src="http://maps.google.com/maps?file=api&amp;v=2&amp;key=ABQIAAAAdjiS7YH6Pzk2NrliO2
b5xxR1ORG5t-vK3TwPKbpNUO2c5sYb4RTmySs_TEFzYvlZrCaYJKlmTzJ5lA"
      type="text/javascript"></script>
```

[57] http://examples.mashupguide.net/ch13/google.map.1.html

```
    <script type="text/javascript">

    //<![CDATA[

    function load() {
      if (GBrowserIsCompatible()) {
        var map = new GMap2(document.getElementById("map"));
        window.map = map
        map.setCenter(new GLatLng(37.4419, -122.1419), 13);
      }
    }

    //]]>
    </script>
  </head>

  <body onload="load()" onunload="GUnload()">
    <div id="map" style="width: 500px; height: 300px"></div>
  </body>
</html>
```

6. Invoke the Javascript shell for your map, by hitting the Javascript shell bookmarklet in the context of your map. Type in the following code fragments and see what happens. (Note that another approach is to modify your code directly with these code fragments and reload your page.) One can correlate the actions to the documentation for v2 of the Google Maps API.[58]

[58] http://www.google.com/apis/maps/documentation/reference.html#GMap2

To return the current zoom level of the map (which goes from 0 to 17, 17 for the most detailed) (the reponse from the Javascript shell is shown right :

```
map.getZoom()
13
```

To obtain the (latitude, longitude) of the center of the map:

```
map.getCenter()
(37.4419, -122.1419)
```

To center the map around the Campanile for UC Berkeley.

```
map.setCenter(new GLatLng(37.872035,-122.257844), 13);
```

You can pan to that location instead:

```
map.panTo(new GLatLng(37.872035,-122.257844));
```

To add a small map control (to control the zoom level) and the

```
map.addControl(new GSmallMapControl());
map.addControl(new GMapTypeControl());
```

To GMap keyboard navigation on, use:

```
window.kh = new GKeyboardHandler(map);
[object Object]
```

To fully zoomed out the map

```
map.setZoom(0)
```

To zoom in all the way: (which may go from 15-17)

```
map.setZoom(17)
```

To set the variable `maptypes` to an array holding 3 objects:

```
maptypes = map.getMapTypes()
[object Object],[object Object],[object Object]
```

To get the name of the first entry in maptypes:

```
map.getMapTypes()[0].getName()[59]
Map
To get the current map type, you can get the object and the name of that type object
map.getCurrentMapType()
[object Object]
map.getCurrentMapType().getName()
Map
```

To set the maptype to "satellite":

```
map.setMapType(maptypes[1]);
```

You can zoom one level in and out if you are not already at the max or min zoom levels:

```
map.zoomIn();
map.zoomOut();
```

To make an overlay, try this

```
point = new GLatLng (37.87309185260284, -122.25508689880371);
(37.87309185260284, -122.25508689880371)
marker = new GMarker(point);
[object Object]
map.addOverlay(marker);
```

To make something happen when you click on the marker:

```
GEvent.addListener(marker, 'click', function() { marker.openInfoWindowHtml('hello');
});
[object Object]
```

[59] 1 corresponds to "Satellite", while 2 corresponds to "Hybrid" map type

There are many more things to explore, such as polylines and overlays and draggable points. To learn more, I certainly recommend the Google Maps API: Introduction[60] Note the assumed indicates that "this documentation is designed for people familiar with Javascript programming and object-oriented programming concepts. You should also be familiar with Google Maps from a user's point of view."

Yahoo Maps API

The Yahoo! maps can be programmed with its API.The core documentation is to found at the Yahoo website: Yahoo! Maps Web Services - Introducing the Yahoo! Maps APIs. [61] As I describe in previous sections, the Simple API[62] is useful to get started with because the declarative approach does not involve any Javascript programming, but only creating the right XML file. Note: I am not covering the Flash APIs for Yahoo maps[63] here; they look cool and are worth looking at, especially for cases where Flash might be superior to a DHTML approach.

In this section, I will introduce you a step-by-step introduction to the AJAX APIs for the Yahoo maps.[64]

1. Apply for a Yahoo application key.[65] We are told not to use the appid for the example code on the Yahoo documentation (`YahooDemo`). The one I have registered for mashupguid.net is `.xKOcvbV34GBpqs2hG6TIO6BFKORhypV3TKQ7mWIsZXDpru04AaZMvbYa_Dv`

2. Copy and paste the following code for your website:[66]

```
<html>
  <head>
    <script type="text/javascript"
src="http://api.maps.yahoo.com/ajaxymap?v=3.0&appid=.xKOcvbV34GBpqs2hG6TIO6BFKORhypV
3TKQ7mWIsZXDpru04AaZMvbYa_Dv "></script>
    <style type="text/css">
      #mapContainer {
      height: 500px;
      width: 80%;
      }
    </style>
  </head>
<body>
  <div id="mapContainer"></div>
```

60 http://www.google.com/apis/maps/documentation/#Introduction

61 http://developer.yahoo.net/maps/

62 http://developer.yahoo.net/maps/simple/index.html

63 http://developer.yahoo.net/maps/flash/index.html

64 http://developer.yahoo.net/maps/ajax/index.html

65 https://developer.yahoo.com/wsregapp/index.php

66 I have my code at http http://examples.mashupguide.net/ch13/yahoo.map.simple.1.html

```
  <script type="text/javascript">
    // Create a lat/lon object
    var myPoint = new YGeoPoint(37.4041960114344,-122.008194923401);
    // Create a map object
    var map = new YMap(document.getElementById('mapContainer'));
    // Display the map centered on a latitude and longitude
    map.drawZoomAndCenter(myPoint, 3);

    // Add map type control
    map.addTypeControl();

    // Set map type to either of: YAHOO_MAP_SAT YAHOO_MAP_HYB YAHOO_MAP_REG
    map.setMapType(YAHOO_MAP_SAT);

    //Get valid map types, returns array [YAHOO_MAP_REG, YAHOO_MAP_SAT,
YAHOO_MAP_HYB]
    var myMapTypes = map.getMapTypes();
  </script>
</body>
</html>
```

3. Invoke the Javascript shell for your map, by hitting the Javascript shell bookmarklet in the context of your map. Type in the following code fragments and see what happens. (If you don't use your own map, try the javascript sell on the example on the Yahoo website)[67].

   * to get the zoom level:

```
map.getZoomLevel()
3
```

   * to get the center location, specifically, the latitude and longitude.

```
map.getCenterLatLon()
[object Object]
props(map.getCenterLatLon())
Fields: Lat, Lon
Methods of prototype: distance, equal, getRad, greater, middle, pointDiff,
setgeobox, valid
map.getCenterLatLon().Lat
37.4041960114344
map.getCenterLatLon().Lon
-122.008194923401
```

   * to set the zoom level (15 for largest scale, 1 for most zoomed in.

```
map.setZoomLevel(15)
map.setZoomLevel(1)
```

   * to move the map to a new center (in this case, the UC Berkeley campus:

[67] http://developer.yahoo.com/maps/ajax/V3/ajaxexample1.html

```
p = new YGeoPoint(37.87309185260284, -122.25508689880371)
[object Object]
map.panToLatLon(p)
```

* to add some navigation and zoom controls

```
map.addPanControl();
map.addZoomLong();
```

* to add a marker, labeled 'H'

```
marker = new YMarker(p);
[object Object]
marker.addLabel("H");
map.addOverlay(marker);
```

* to make a click event invoke a popup:

```
function onSmartWinEvent() {var words = "Yeah Yahoo maps!";
marker.openSmartWindow(words); }
YEvent.Capture(marker, EventsList.MouseClick, onSmartWinEvent);
```

* to add a marker to a specific address, in this case, 2195 Hearst Ave, you can use either of the following three alternatives:

```
map.addMarker("2195 Hearst Ave, Berkeley, CA");
```

or

```
map.addOverlay(new YMarker("2195 Hearst Ave, Berkeley, CA"));
```

or

```
marker = new YMarker("2195 Hearst Ave, Berkeley, CA");
marker.addLabel("2195 Hearst");
map.addOverlay(marker);
marker.reLabel("<b>hello</b>");
```

These examples do not exhaust the Yahoo AJAX API, which has features such as polylines and more complete overlay functionality.

Microsoft's Windows Live Local/Virtual Earth

Microsoft's offering in online maps has gone under quite a few names (Windows Live Local, and Virtual Earth). From the Wikipedia entry on Windows Live Local, we read:

> ***Windows Live Local*** *is a free web map server provided as a part of Microsoft's Windows Live online applications services suite. At http://local.live.com (http://maps.live.com or http://virtualearth.com), it offers street maps, satellite imagery, driving directions, and traffic information.*

My read of the situation is that Windows Live Local is the official name of Microsoft' next generation maps, and the technology behind it is "Virtual Earth".[68] None of this technology is to be confused with MSN Maps.[69]

The Virtual Earth Map Control is an AJAX widget, well documented at the following locations:

* the official central place for the Microsoft Virtual Earth docs[70]
* The Virtual Earth Interactive SDK[71] is a great place to learn about Virtual Earth because it combines a live demo with relevant source code and links to the reference documentation.

Walk-thru:

1. Copy and paste the following code (this is the simplest piece of code given at the Virtual Earth Interactive SDK):[72]

```
<html>
   <head>
      <title></title>
      <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
      <script
src="http://dev.virtualearth.net/mapcontrol/v5/mapcontrol.js"></script>
      <script>
      var map = null;

      function GetMap()
      {
         map = new VEMap('myMap');
         map.LoadMap();
      }
      </script>
   </head>
   <body onload="GetMap();">
      <div id='myMap' style="position:relative; width:400px; height:400px;"></div>
   </body>
</html>
```

68 See http://radar.oreilly.com/archives/2005/12/virtual_earth_becomes_windows.html and note that http://virtualearth.com redirects to http://local.live.com

69 http://mappoint.msn.com/

70 http://dev.live.com/virtualearth

71 http://dev.live.com/virtualearth/sdk/

72 http://examples.mashupguide.net/ch13/ve.map.1.html

2. Invoke the Javascript shell for your map, by hitting the Javascript shell bookmarklet in the context of your map. Type in the following code fragments and see what happens. [73] (Note in addition to applying the javascript shell to this special embedded map, I can get a map object of a link in local.live.com.[74])

   * to get the zoom level:

```
map.GetZoomLevel()
4
```

   * to zoom in and out:

```
map.ZoomIn();
```

and

```
map.ZoomOut();

map.GetMapStyle()
r
```

   * to set the map style (a (aerial), r (road), h (hybrid), or o (birds-eye). are valid options):

```
map.SetMapStyle('a');
```

Noteworthy aspects of Virtual Earth:

* There are 3D renditions of the map available if you are running Internet Explorer version 6 or 7 in Windows and have a 3-D add-ons installed. (check this)
* the virtual Earth API (version 5) does include support for the three-dimensional views. However, since the three-dimensional functionality is available only in Internet Explorer, we would not be able to see that in action using the JavaScript shell, which only works in Firefox.

It would be nice to do a bit of coverage of the 3D API functionality in VE. Let me quote a passage the Virtual Earth blog:[75]

> *What did I say the best part was? I changed my mind. the best part of this release is that it is supported with our javascript API. you read that correctly! if you can write the 5 lines of javascript to manipulate Virtual Earth in 2d, it is the*

[73] In contrast to some other major online map services (such as maps.google.com and maps.yahoo.com), it is possible to get access to the map object in local.live.com.

[74] http://tinyurl.com/y9jrsx or http://local.live.com/default.aspx?v=2&ss=yp.berkeley%2c%20ca&cp=38.889801~-77.0059&style=r&lvl=15&tilt=-90&dir=0&alt=-1000&scene=182507&sp=yp.YN120x2223273~Point.qgfg2z8kgjqr_E%20Capitol%20St%20NE%20%26%201st%20St%20NE%2c%20Washington%2c%20DC%2020003%2c%20United%20States___

[75] http://virtualearth.spaces.live.com/Blog/cns!2BBC66E99FDCDB98!7573.entry

> *same code to instantiate the map control and switch to 3d. Here's a snippet of code if you want to try it:*

And the code snippet is:

```
<script type="text/javascript">
  var map = null;
  function OnPageLoad() {
    map = new VEMap('map');
    map.LoadMap(new VELatLong(42.35822, -71.05309), 19, 'h', false,VEMapMode.Mode3D
);
  }
</script>
```

Mapquest maps and the Mapquest OpenAPI

Although MapQuest is one of the most popular of online map services in the United States, it is relatively new to offering an API to its maps (MapQuest OpenAPI).[76] My overall impression is that the community around the API is small, uptake has been slow, and the maps themselves are not quite as fluid and powerful as its competitors. Nonetheless, because MapQuest is a popular brand, with AOL as its current owners, I believe it worthwhile showing how to get started with the API.

Let's walk through how to create a simple MapQuest map:

1. Sign up for a key by registering your application -- or if you have already signed up for an account, log in.[77] The trickiest part of getting started is getting `referer` right. Note warning: "A key is valid for a single "referer". If you sign up for the referer http://www.mywebsite.com/mymaps/, the key you get will be good for all URLs within the http://www.mywebsite.com/mymaps/ referer."

   When you get passed the account set up stage (which allows you to enter a single referer), you can use the account manager[78] to enter other referers so that you use mapquest maps served under more than one URL. "By entering a referer here, your are limiting the usage of your OpenAPI key to only these areas of the web. If you add a wildcard * symbol to the end of the referer, then any page and any directory, sub-directory, or part of a domain name will also work."

Insert 858Xf1301.tif

Figure 13-01. ???

[76] http://www.mapquest.com/openapi/

[77] https://trc.mapquest.com/

[78] https://trc.mapquest.com/?action=accountmanager

2. Go to the URL given in your registration email. "You will receive two emails in response to your request. The first will serve as a record of your request and acceptance of the terms of the Evaluation Agreement. Once you accept the terms and verify your registration, a second email will follow which provides all the information needed to begin using MapQuest OpenAPI. Your requests will stay open for two days....You will receive information from the openAPI@mapquest.com email address. We suggest you allow this address through any spam filter you may have." My URL was http://company.mapquest.com:80/reg?action=confirm&account=51d8c26719be178c06c6538a011e4aff

3. When you click on that URL, you will get a followup email that has the 1) password for your account and 2) the access key. Keep the password secret. You will be using the key in your maps. The key for raymondyee is `mjtd%7Clu6t29ubn9%2C25%3Do5-gaa2d`. If you don't get this right, you will get an error message:

Insert 858Xf1302.tif

Figure 13-02. ???

4. Copy and paste the following code into a file in your directory[79]::

```
  <html>
    <head>
      <script
src="http://web.openapi.mapquest.com/oapi/transaction?request=script&key=mjtd%7Clu6t29
ubn9%2C25%3Do5-gaa2d"
      type="text/javascript"></script>
      <title>Mapquest OpenAPI Sample - Single Map</title>
    </head>
    <body style="background-color:rgb(244, 246, 229)" id='body' >
      <div id='myMap' class='myMap' style="height:550;width:550"> </div>
      <script type="text/javascript">
        var map = new MQMap("myMap");

        var loc1 = new MQLocation();
        loc1.setName("Empire State Building");
        loc1.setAddress("350 5th Ave");
        loc1.setCity("New York");
        loc1.setStateProvince("NY");
        loc1.setPostalCode("10118");
        loc1.setIconId(11);
        map.locations.add(loc1);

        var loc2 = new MQLocation();
```

[79] http://examples.mashupguide.net/ch13/mapquest.1.html

```
          loc2.setName("United Nations");
          loc2.setAddress("760 United Nations Plaza");
          loc2.setCity("New York");
          loc2.setStateProvince("NY");
          loc2.setPostalCode("10017");
          loc2.setIconId(14);
          map.locations.add(loc2);

          var loc3 = new MQLocation();
          loc3.setName("New York Public Library");
          loc3.setAddress("5th Avenue and 42nd Street");
          loc3.setCity("New York");
          loc3.setStateProvince("NY");
          loc3.setPostalCode("10036");
          loc3.setIconId(23);
          map.locations.add(loc3);

          map.setRolloverPopups(true);//To enable the popups

          map.getMap();

        </script>
      </body>
    </html>
```

5. Now invoke the Javascript shell to learn about the various aspects of the MapQuest OpenAPI.

`map` is the map object.

* to get the dimension of the map in pixels:

```
map.getMapSize().getHeight()
492
map.getMapSize().getWidth()
466
```

* to get the style (options: bw, classic, default, style5, european)

```
map.getMapStyle()
style5
```

* to get the latitude and longitude of the center of the map

```
map.getMapCenterLL().getLat()
40.734978
map.getMapCenterLL().getLng()
-73.981985
```

* to change the style of the map (you need to invoke getMap() to render the map)

```
map.setMapStyle('european');
map.getMap()
```

* to zoom in and out:

```
map.zoomOut()
map.zoomIn()
```

* to set the zoom level (1 is most zoomed out; 10 is most zoomed in) (weird zoom level functions!):

```
map.zoomLevel1()
map.zoomLevel10()
```

* to turn the rollover off and on:

```
map.setRolloverPopups(false)
map.setRolloverPopups(true)
```

* to get at the locations and their properties, for instance, to the name of the first location:

```
map.locations.getAt(0).getName()
Empire State Building
```

* to override how info ballons are displayed:

```
function dispInfo(x,y,locColl ){var size = locColl.getSize(); var htmlStr = ""; for
(var i=0; i < size ;i++){ var loc = locColl.getAt(i); htmlStr = htmlStr + "<p>" +
"DispInfo: " + loc.getName() + "<br/>&nbsp;&nbsp;&nbsp;" + loc.getCity() +
"<br/>&nbsp;&nbsp;&nbsp;" + "Heavy Traffic" + "</p><p>currentX:"+ loc.getX()+"
currentY: " + loc.getY()+ "</p><br/>" ; } return (htmlStr); }

map.addListener("rollover", "dispInfo");
```

Insert 858Xf1303.tif

Figure 13-03. ???

Driving Directions from MapQuest

One of the nice features of the MapQuest OpenAPI is the easy creation of driving directions. Is it unique?[80] Let's demonstrate how you can do that. Copy and paste the following code:

[80] Yahoo (http://developer.yahoo.com/maps/ajax/V3.4/reference.html) and Google (http://www.google.com/apis/maps/documentation/#Routing_Local_Etc) don't provide afree geo-routing API. The Virtual Earth widget does have routing capability (http://dev.live.com/virtualearth/sdk/Ref/HTML/M_Namespace_VEMap_GetRoute.htm). Maybe

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>
   <script
src="http://web.openapi.mapquest.com/oapi/transaction?request=script&key=mjtd%7Clu6t
29ubn9%2C25%3Do5-gaa2d"  type="text/javascript"></script>
<!--
   <link rel="stylesheet" href="./css/routecontent.css" type="text/css"
media="screen, projection, print" />
-->
   <title>
      OpenAPI Sample - Routing
   </title>
</head>
<body id='body'>
       <div id='container' class='mqContainer'
style='height:400px;width:600px'></div>
  <script type="text/javascript">

    var mqRoute = null;
    mqRoute = new MQRoute("container");

    var thumbSize = new MQSize();
    thumbSize.setHeight(150);
    thumbSize.setWidth(300);

    var overviewSize = new MQSize();
    overviewSize.setHeight(400);
    overviewSize.setWidth(600);

    mqRoute.primaryMapSize = overviewSize;

    mqRoute.origin.setAddress("Fifth Avenue and 42nd Street");
    mqRoute.origin.setCity("New York");
    mqRoute.origin.setStateProvince("NY");
    mqRoute.origin.setPostalCode("10036");
    mqRoute.origin.setName("New York Public Library");

    mqRoute.destination.setAddress("United Nations");
    mqRoute.destination.setCity("New York");
    mqRoute.destination.setStateProvince("NY");
    mqRoute.destination.setPostalCode("10017");
    mqRoute.destination.setName("United Nations");

    mqRoute.doRoute("routeReturn");
</script>
</body>
</html>
```

Note what this code does: when the page loads, an HTML div with an id of container, after which the script is run to rewrite the div with map route information, specifically, starting from the origin of the New York Public Library and arriving at the United Nations.

Using this code segment as a basic component, you can then modify it to take an origin or a destination and dynamically return routing information.

How to learn more about the MapQuest OpenAPI

* Download the zip file of examples.[81]
* Read the OpenAPI User Guide.[82]
* Consult the OpenAPI documentation.[83]

The example works -- but there is ONE BIG PROBLEM -- the dots show up in the wrong place! I will post about this problem to ask why. The Empire State building, whose address is 350 5th Avenue, NY, NY is not geocoded properly. If you explicitly run the MapQuest geocoder[84], you will find that the geocoder thinks that 350 5th Ave is 350 E 5th Street! This is odd since mapquest.com doesn't have a problem.[85]

Geocoding

A very common task in using online maps is to geocode addresses -- that is, to convert street addresses to the corresponding latitude and longitude. In this section, I will walk through the basics of geocoding in each of of the Yahoo, Google, Virtual Earth, and MapQuest maps. (There are subtleties that I won't go in detail about: the precision and accuracy of the APIs, dealing with ambiguities in the addresses, which geocoder is best for a given geographic location.)

[81] https://trc.mapquest.com/download.adp?action=download&file=/samples/MQOpenAPI_samples.zip

[82] https://trc.mapquest.com/download.adp?action=download&file=/docs/oapi/OpenAPI-DevGuide.pdf -- requires a login.

[83] https://trc.mapquest.com/content/oapi/docs/index.html

[84] http://web.openapi.mapquest.com/oapi/transaction?transaction=geocode&address=350+5th+Avenue&city=New+York&stateProvince=NY&key=mjtd%7Clu6t29ubn9%2C25%3Do5-gaa2d

[85] http://www.mapquest.com/maps/map.adp?address=350%205th%20Ave&city=New%20York&state=NY&zipcode=10118%2d0110&country=US&title=%3cb%20class%3d%22fn%20org%22%3e350%205th%20Ave%3c%2fb%3e%3cbr%20%2f%3e%20%3cspan%20style%3d%22display%3ainline%3bmargin%2dbottom%3a0px%3b%22%20class%3d%22locality%22%3eNew%20York%3c%2fspan%3e%2c%20%3cspan%20style%3d%22display%3ainline%3bmargin%2dbottom%3a0px%3b%22%20class%3d%22region%22%3eNY%3c%2fspan%3e%20%3cspan%20style%3d%22display%3ainline%3bmargin%2dbottom%3a0px%3b%22%20class%3d%22postal%2dcode%22%3e10118%2d0110%3c%2fspan%3e%2c%20%20%3cspan%20style%3d%22display%3ainline%3bmargin%2dbottom%3a0px%3b%22%20class%3d%22country%2dname%22%3eUS%3c%2fspan%3e%3c%2fspan%3e&cid=lfmaplink2&name=&dtype=s

Yahoo Maps

Yahoo provides a REST geocoding method,[86] whose baseURL is `http://api.local.yahoo.com/MapsService/V1/geocode` and parameters include the `appid` to identify your application and two ways of identifying the address:

1. a combination of `street`, `city`, `state`, and `zip`
2. `location`, "free text" which is one string that holds a combination of `street`, `city`, `state`, and `zip` The `location` string has priority for determining the placement of

Let's use these two methods to geocode the location of the Empire State Building, 350 5th Avenue, New York, New York, 10118:

1. using `street`, `city`, and `state`,[87] returns:

```
<?xml version="1.0"?>
<ResultSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="urn:yahoo:maps" xsi:schemaLocation="urn:yahoo:maps
http://api.local.yahoo.com/MapsService/V1/GeocodeResponse.xsd">
  <Result precision="address">
    <Latitude>40.748434</Latitude>
    <Longitude>-73.984791</Longitude>
    <Address>350 5TH AVE</Address>
    <City>NEW YORK</City>
    <State>NY</State>
    <Zip>10118-0110</Zip>
    <Country>US</Country>
  </Result>
</ResultSet>
```

2. using the `location` parameter alone:[88]

```
<?xml version="1.0"?>
<ResultSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="urn:yahoo:maps" xsi:schemaLocation="urn:yahoo:maps
http://api.local.yahoo.com/MapsService/V1/GeocodeResponse.xsd">
  <Result precision="address">
    <Latitude>40.748434</Latitude>
    <Longitude>-73.984791</Longitude>
    <Address>350 5TH AVE</Address>
    <City>NEW YORK</City>
    <State>NY</State>
```

---

[86] http://developer.yahoo.com/maps/rest/V1/geocode.html

[87] http://api.local.yahoo.com/MapsService/V1/geocode?appid=raymondyee.net&street=350+5th+Ave&city=New+York&state=NY

[88] http://api.local.yahoo.com/MapsService/V1/geocode?appid=raymondyee.net&location=350+5th+Ave,+New+York,+NY

```
    <Zip>10118-0110</Zip>
    <Country>US</Country>
  </Result>
</ResultSet>
```

Note the following characteristics of the output:

1. You get the same results with the two methods in this case.

2. You can compare the output address and the input address to make sure that the geocoder is interpreting the address the way you think it should be.

3. The default output format is xml when no `output` parameter is unspecified.

4. You get the latitude and longitude in the Latitude and Longitude element, respectively.

A good way to see how the API behaves is to try out various parameters. See what happens in the following cases:

* when you specify only `city=Berkeley` to get several results corresponding to the cities that go by the name of Berkeley[89]
* when you use the `output=php` option to get "serialized php"[90]
* when you enter a non-existent street address for a given city[91]
* when you use try to get json output, which used to be provided by the API.[92]

Google Geocoder

Google provides a REST interface and geocoding functionality through Javascript.[93]

Looking first at the REST method, whose base URL is http://maps.google.com/maps/geo? and whose parameters are

* `q`: The address to geocode
* `key`: your API key[94]

---

[89] http://api.local.yahoo.com/MapsService/V1/geocode?appid=raymondyee.net&city=Berkeley

[90] http://api.local.yahoo.com/MapsService/V1/geocode?appid=raymondyee.net&location=350+5th+Ave,+New+York,+NY&output=php

[91] http://api.local.yahoo.com/MapsService/V1/geocode?appid=raymondyee.net&location=350000+main+Street,+Berkeley,+CA

[92] It seems that that the Yahoo geocoder used to support json output but no longer. You see signs of json output at http://www.xml.com/lpt/a/1636, http://tech.groups.yahoo.com/group/yws-maps/message/1760 and http://www.theurer.cc/blog/2005/12/15/web-services-json-dump-your-proxy

[93] http://www.google.com/apis/maps/documentation/#Geocoding_Examples

* `output`: the format of the output -- one of `xml`, `kml`, `csv`, or `json`.

Let's look at some example output:

* `xml` output for 350 5th Ave, New York, NY:[95]

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.0">
  <Response>
    <name>350 5th Ave, New York, NY</name>
    <Status>
      <code>200</code>
      <request>geocode</request>
    </Status>
    <Placemark>
      <address>350 5th Ave, New York, NY 10001, USA</address>
      <AddressDetails Accuracy="8" xmlns="urn:oasis:names:tc:ciq:xsdschema:xAL:2.0">
        <Country>
          <CountryNameCode>US</CountryNameCode>
          <AdministrativeArea>
            <AdministrativeAreaName>NY</AdministrativeAreaName>
            <SubAdministrativeArea>
              <SubAdministrativeAreaName>New York</SubAdministrativeAreaName>
              <Locality>
                <LocalityName>New York</LocalityName>
                <Thoroughfare>
                  <ThoroughfareName>350 5th Ave</ThoroughfareName>
                </Thoroughfare>
                <PostalCode>
                  <PostalCodeNumber>10001</PostalCodeNumber>
                </PostalCode>
              </Locality>
            </SubAdministrativeArea>
          </AdministrativeArea>
        </Country>
      </AddressDetails>
      <Point>
        <coordinates>-73.984653,40.748324,0</coordinates>
      </Point>
    </Placemark>
  </Response>
</kml>
```

* `json` output for convenient use in javascript:[96]

94

ABQIAAAAdjiS7YH6Pzk2Nrli02b5xxRv9hyeHeDQ2dFc2Zk8KJ_0umn6dxQBVteh7bB5CLZbwnVET0bgcyGM6g is the key for http://examples.mashupguide.net/

95

http://maps.google.com/maps/geo?q=350+5th+Ave,+New+York,+NY&output=xml&key=ABQIAAAAdjiS7YH6Pzk2Nrli02b5xxRv9hyeHeDQ2dFc2Zk8KJ_0umn6dxQBVteh7bB5CLZbwnVET0bgcyGM6g

```
{"name":"350 5th Ave, New York,
NY","Status":{"code":200,"request":"geocode"},"Placemark":[{"address":"350 5th Ave,
New York, NY 10001,
USA","AddressDetails":{"Country":{"CountryNameCode":"US","AdministrativeArea":{"Admi
nistrativeAreaName":"NY","SubAdministrativeArea":{"SubAdministrativeAreaName":"New
York","Locality":{"LocalityName":"New York","Thoroughfare":{"ThoroughfareName":"350
5th Ave"},"PostalCode":{"PostalCodeNumber":"10001"}}}}},"Accuracy":
8},"Point":{"coordinates":[-73.984653,40.748324,0]}}]}
```

* invoking the `kml` output[97] might end up lauching Google Earth and navigating to the position. A portable way to render the kml output is to feed it into Google maps.[98] (That is, you enter the URL of the kml file:

```
http://maps.google.com/maps/geo?q=350+5th+Ave,+New+York,+NY&output=kml&key=ABQIAAAAd
jiS7YH6Pzk2Nrli02b5xxRv9hyeHeDQ2dFc2Zk8KJ_0umn6dxQBVteh7bB5CLZbwnVET0bgcyGM6g
```

into the Google Maps search box.

As for the Javascript methods available in the Google mapping system, consult the documentation provided by Google,[99] which points to using the `GClientGeocoder` object. You can use the Javascript shell to see this object in action:

1. Bring up the simple Google map from the previous section.[100]

2. In the Javascript shell, invoke the following commands to illustrate the use of `GClientGeocoder.getLatLng`, which takes in an `address` and a callback function, which in turn takes the `point` geocoded from address:

```
address = "350 5th Ave, New York, NY"
350 5th Ave, New York, NY
geocoder = new GClientGeocoder();
[object Object]
geocoder.getLatLng( address, function(point) { if (!point) { alert(address + " not
found");} else { map.setCenter(point, 13); var marker = new GMarker(point);
map.addOverlay(marker); marker.openInfoWindowHtml(address); } } );
```

3. You will then see the Google Maps add an overlay marking 350 5th Ave, NY, NY.

---

96 http://maps.google.com/maps/geo?q=350+5th+Ave,+New+York,+NY&output=json&key=ABQIAAAAdji S7YH6Pzk2Nrli02b5xxRv9hyeHeDQ2dFc2Zk8KJ_0umn6dxQBVteh7bB5CLZbwnVET0bgcyGM6g

97 http://maps.google.com/maps/geo?q=350+5th+Ave,+New+York,+NY&output=kml&key=ABQIAAAAdji S7YH6Pzk2Nrli02b5xxRv9hyeHeDQ2dFc2Zk8KJ_0umn6dxQBVteh7bB5CLZbwnVET0bgcyGM6g

98 http://maps.google.com/maps?f=q&hl=en&q=http://maps.google.com/maps/geo%3Fq%3D350%2B5th%2 BAve,%2BNew%2BYork,%2BNY%26output%3Dkml%26key%3DABQIAAAAdjiS7YH6Pzk2Nrli02b5x xRv9hyeHeDQ2dFc2Zk8KJ_0umn6dxQBVteh7bB5CLZbwnVET0bgcyGM6g+&ie=UTF8&z=17&om=1

99 http://www.google.com/apis/maps/documentation/#Geocoding_JavaScript

100 http://raymondyee.dreamhosters.com/book/map/google.map.1.html

Insert 858Xf1304.tif

Figure 13-04. ???

Virtual earth

Virtual Earth provides geocoding functionality in the `VEMap.FindLocation` method[101], which takes two parameters, a location string and a callback function. Let's illustrate this at work with the Javascript shell:

1. Bring up the basic Virtual Earth example.[102]

2. Let's use the address of the New York Public Library (5th Avenue and 42nd Street, New York, NY) as an example. Let's also create a callback function which pops up an alert with the latitude/longitude of the found location:

```
function onFoundResults(e){

  html = "";
  for (x=0; x<e.length; x++) {
    html = html + e[x].LatLong + "";
  }
  alert (html);

}
```

3. In the Javascript shell, type:

```
address = "5th Avenue and 42nd Street, New York, NY"
5th Avenue and 42nd Street, New York, NY
function onFoundResults(e){ html = "";for (x=0; x<e.length; x++) { html = html +
e[x].LatLong + "";} alert (html); }
map.FindLocation(address,onFoundResults)
```

4. You will see an alert that pops up the latitude and longitude of the address.

MapQuest

Two services are provided by MapQuest:

* REST call
* Javascript API

Let me provide examples:

* REST call to encode 350 5th Ave, New York, NY[103]

[101] http://dev.live.com/virtualearth/sdk/ref/HTML/M_Namespace_VEMap_FindLocation.htm

[102] http://raymondyee.dreamhosters.com/book/map/ve.map.1.html

* to do the Javascript encoding, we need to use the `MQGeocode` object.[104]

Let's use the javascript shell to look at MQGeocode in action:

1. Bring up a simple MapQuest map.[105]
2. Type the following to do a geocoding of 350 5th Ave, New York, NY:

```
map
[object Object]
geocoder = new MQGeocode()
[object Object]
loc = new MQLocation()
[object Object]
loc.setAddress("350 5th Ave")
loc.setCity("New York")
loc.setStateProvince("NY")
function onGeocode(g) { html = ""; for (x=0; x<g.locations.getSize(); x++) { html =
html + x + ":" + g.locations.getAt(x).getLatitude() + "," +
g.locations.getAt(x).getLongitude(); } alert (html); }
geocoder.doGeocode(loc,"onGeocode")
```

3. This results in a javascript alert with the latitude and longitude of the address.

Other Geocoding services

The Google, Yahoo, Microsoft and MapQuest geocoders are specialized for American addresses (and perhaps Canadian addresses too). It would be helpful to find geocoders to work well for non-American addresses. One example is http://map24.com, which seems to have strength for a geocoding European addresses. Here are some other

Google Earth

Google Earth[106] is a "virtual globe," which means that it is a desktop environment that simulates the three-dimensional aspects of the earth. It runs on Windows, Mac OS X, and Linux. Google earth is a very cool application, rightfully described as immersive. I won't be surprised if it becomes a dominant platform for geo-data sharing. [107]

Google Earth is also a great matchup platform. What makes it so?

103
http://web.openapi.mapquest.com/oapi/transaction?transaction=geocode&address=350+5th+Avenue&city=New+York&stateProvince=NY&key=mjtd%7Clu6t29ubn9%2C25%3Do5-gaa2d

104 https://trc.mapquest.com/content/oapi/docs/content/_0abDUGtKEdqjTrv0zWM0cg_root.html

105 http://examples.mashupguide.net/ch13/mapquest.1.html

106 http://earth.google.com/

107 http://www.technologyreview.com/read_article.aspx?ch=specialsections&sc=personal&id=17537 makes the argument that Google Earth will be exactly that dominant platform.

* the three-dimensional space is a planet is in organizing framework that is easy to understand -- everyone knows his or her place in the world to speak
* KML -- the XML data format for getting data in and out of Google Earth is easy to read and to write
* there are other APIs to Google Earth, including a COM interface in Windows and an applescript interface in Mac OS X.

KML and the Google Earth interface

The following is a breakdown of skills needed to produce KML out of Google Earth without any programming. I still need to turn this into a step-by-step how to.

The first let's look at Google Earth as an end user as as a way into the question of how to mash up. Google Earth. I will use the scenario of planning a trip and seeing how I can info in and out of GE. In order to produce KML from just using Google Earth, it's useful to have the following skills:

* It would be important for you to be comfortable with typing in addresses or business names, causing the GE interface to go to those places
* I want to show you how to make collections in the Google Earth interface, how to change the properties of individual items, including the latitude, longitude, icon, view of the item.
* Once you have a collection in Google Earth, then I want you to be to be able to get KML corresponding to the collection.[108] With the KML in hand and an understanding of what a collection looks like in Google Earth, you are in a good position to not only to read KML, but also write to write KML.

Let's look at an excerpt of KML produced out of sights I want to see recently are in a trip to San Diego:[109]

It also be very healthful to have a screenshot of Google Earth that corresponds to this KML.

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
  <Document>
    <name>San.Diego.Destinations.kml</name>
[...]
    <Folder>
      <name>San Diego destinations</name>
      <open>1</open>
```

[108] To start with, you will need to know where to find the KML on your hard disk: C:\Documents and Settings\Administrator\Application Data\Google\GoogleEarth\myplaces.kml (Win32) and ~/Library/Google Earth/myplaces.kml (OS X).

[109] http://raymondyee.dreamhosters.com/book/map/San.Diego.Destinations.kml

```
      <Placemark>
        <name>Cafe Lulu</name>
        <open>1</open>
        <address>419 F St&lt;br/&gt;San Diego, CA 92101</address>
        <Snippet maxLines="2"><![CDATA[419 F St, San Diego, CA 92101<br/>(619) 238-
0114]]></Snippet>
        <styleUrl>#default+nicon=0x500+hicon=0x510</styleUrl>
        <Point>
          <coordinates>-117.160821,32.713613,0</coordinates>
        </Point>
      </Placemark>
      <Placemark>
        <name>Filippi&apos;s Pizza Grotto: Chula Vista</name>
        <open>1</open>
        <address>82 Broadway&lt;br/&gt;Chula Vista, CA 91910</address>
        <Snippet maxLines="2"><![CDATA[82 Broadway, Chula Vista, CA 91910<br/>(619)
427-6650]]></Snippet>
        <styleUrl>#default+nicon=0x506+hicon=0x516</styleUrl>
        <Point>
          <coordinates>-117.095068,32.645355,0</coordinates>
        </Point>
      </Placemark>
      <Placemark>
        <name>Filippi&apos;s Pizza Grotto: San Diego</name>
        <open>1</open>
        <address>1747 India St&lt;br/&gt;San Diego, CA 92101</address>
        <Snippet maxLines="2"><![CDATA[1747 India St, San Diego, CA 92101<br/>(619)
232-5094]]></Snippet>
        <styleUrl>#default+nicon=0x502+hicon=0x512</styleUrl>
        <Point>
          <coordinates>-117.168306,32.723253,0</coordinates>
        </Point>
      </Placemark>
      <Placemark>
        <name>Filippi&apos;s Pizza Grotto: Pacific Beach</name>
        <open>1</open>
        <address>962 Garnet Ave&lt;br/&gt;San Diego, CA 92109</address>
        <Snippet maxLines="2"><![CDATA[962 Garnet Ave, San Diego, CA 92109<br/>(858)
483-6222]]></Snippet>
        <styleUrl>#default+nicon=0x500+hicon=0x5101241518</styleUrl>
        <Point>
          <coordinates>-117.252531,32.797183,0</coordinates>
        </Point>
      </Placemark>
    </Folder>
  </Document>
</kml>
```

You can also feed the KML to google maps: to see the destinations on a Google Map.[110] Downside: the captions are not very informative. Is there a way to make them more informative?

The KML language

Although KML at its heart is a simple dialect of XML, Google is steadily adding features to do more and more through KML. To learn more:

* start with the introduction to KML 2.0[111]
* run through a tutorial for KML 2.0.[112]
* then go through the KML to buy one tutorial to see what's new in KML.[113]

You can always consult the KML 2.1 Reference if you are stuck.[114]

Programming Google Earth via COM and AppleScript

It turns out that you can program Google Earth too. In Windows, you can do so through the COM interface[115] and on Mac OS X via Applescript [116].

COM interface

Here's a small sample Python snippet to load the KML of the San Diego trip:

```
import win32com.client
googleEarth =  win32com.client.Dispatch("GoogleEarth.ApplicationGE")
import urllib
f =
urllib.urlopen("http://raymondyee.dreamhosters.com/book/map/San.Diego.Destinations.k
ml")
data = f.read()
f.close()
googleEarth.LoadKmlData(data)
```

A few more code snippets that might be useful to demonstrate what is possible via the COM interface:

```
cl = googleEarth.GetFeatureByName('Cafe Lulu')
googleEarth.SetFeatureView(cl,0.5)  #0.5 is a speed
```

110 http://maps.google.com/maps?f=q&hl=en&q=http://raymondyee.dreamhosters.com/book/map/San.Diego.Destinations.kml

111 http://earth.google.com/kml/kml_intro.html

112 http://earth.google.com/kml/kml_tut.html

113 http://earth.google.com/kml/kml_21tutorial.html

114 http://earth.google.com/kml/kml_tags_21.html

115 http://earth.google.com/comapi/index.html

116 http://www.ogleearth.com/2006/09/google_earth_fo_6.html

```
cl.Highlight #doesn't seem to work he
googleEarth.SetCameraParams( 41.487942634608913, -81.686570904088171, 0.0, 1,
150.00011938469936, 70.000000000947935, -127.30257903894255, 100)
```

Applescript interface to Google Earth

In OS X, we would be using Applescript or appscript. Here are some references:

* Ogle Earth: A blog about Google Earth. « Google Earth for Mac + AppleScript = Geotagger! (and more!)[117]
* The Daily ACK: Google Earth and Applescript[118]
* The Daily ACK: Automating Google Earth[119]

Here's a little code segment in Applescript to get one started:

```
tell application "Google Earth"
  activate
  set viewInfo to (GetViewInfo)
  set dest to {latitude:57.68, longitude:-95.4, distance:1.0E+5,
tilt:90.0, azimuth:180}
  SetViewInfo dest speed 0.1
end tell
```

What does it do? I moves the view to 57.68 N and 95.4 W.

You can program Google Earth using the Applescript framework by using appscript, " a high-level, user-friendly Apple event bridge that allows you to control scriptable Mac OS X applications using ordinary Python scripts."[120] With appscript in place, the following Python script also steers Google Earth in Mac OS X:

```
#!/Library/Frameworks/Python.framework/Versions/Current/bin/pythonw
from appscript import *
ge = app("Google Earth")
#h = ge.GetViewInfo()
h = {k.latitude: 36.510468818615237, k.distance: 5815328.0829986408,
k.azimuth:
10.049582258046936, k.longitude: -78.864908202209094, k.tilt:
3.0293063358608456
e-14}
ge.SetViewInfo(h,speed=0.5)
```

It is surprising to me that these interfaces don't allow one to get get parameters from the Views in MyPlaces. Maybe you have to go read the KML for the MyPlaces off the computer.

---

[117] http://www.ogleearth.com/2006/09/google_earth_fo_6.html

[118] http://www.babilim.co.uk/blog/2006/09/google-earth-and-applescript.html

[119] http://www.babilim.co.uk/blog/2006/11/automating-google-earth.html

[120] http://appscript.sourceforge.net/

Mapstraction and OpenLayers

In this chapter, I cover how to use some of major mapping APIs: Google Maps, Yahoo, Mapquest, and Microsofts's. It would be convenient to be able to not worry about the differences among the maps and easily switch among the various maps. That's the promise of a mapping"abstraction" library such as Mapstraction (http://mapstraction.com). We'll have to see how and whether it is widely used to gauge the library's effectiveness.

Along a different vein is OpenLayers (http://www.openlayers.org/), which is

> *a pure JavaScript library for displaying map data in most modern web browsers, with no server-side dependencies. OpenLayers implements a (still-developing) JavaScript API for building rich web-based geographic applications, similar to the Google Maps and MSN Virtual Earth APIs, with one important difference -- OpenLayers is Free Software, developed for and by the Open Source software community.*

You can try out OpenLayers in FlashEarth. Go to the site and select OpenLayers. You might have to zoom out sufficiently to see any tiles (e.g., http://www.flashearth.com/?lat=38.417308&lon=-122.271821&z=9.9&r=0&src=ol) You can also check out other examples in the OpenLayers gallery: http://www.openlayers.org/gallery/

Culminating Example(s)

It seems to me that in addition to covering all the pieces of maps in detail, that now could be helpful to have a few integrating examples. At this point, no sketch a few possibilities, and then fill in the one we choose.

Geocode a few addresses and generate KML to feed to Google Earth

This would be a fairly straightforward mashup of one of the geocoders in KML. I want an array of names, addresses, and possible associated HTML. We can do the geocoding server-side or client-side (Javascript). Maybe we can mix-and-match geocoding services and mapping services.

Showing Flickr pictures in Google Earth

I think that showing a simple translation of Flickr pictures in Google Earth might be a good example to show here or elsewhere in the book. It might be possible to leave an extended example in which Google Earth becomes a dynamic interface to Flickr pictures. That is, as we changed the view in Google Earth, and go back to Flickr to get pictures within them you. The sample code here using NetworkLink builds a skeleton for part of what we need.

The example I want to write for the book is a simple display pictures in Google Earth using the network links in KML. If we wanted to get fancy, we would apply some data clustering and draw upon some existing clustering code.[121]

A good demo would be a Google Earth interface to the Flickr geotagged photos. You have to make use of the `NetworkLink` element in KML. V1 T3here 8.Mnk lttple eyo ofshow

mfenralVLi95(e)]TJ0.0004 Tc47.14203 Td[wN(et9(wg)26workLik.km)12(lh)]TJEMC /P /MCID364 BDC 0 Tc64482 0 064482 9084.84001 Tm((1

sdn2.mfcrosoft.com/

ondyee.dreamhosars.com/bgoN

```
$timesnap = date("H:i:s");

// split the client's BBOX return by commas and spaces to obtain an array of
coordinat a es
$coords = preg_split('/,|\s/', $_GET["BBOX"]);

// for clarity, place each coordinate into a clearly marked bottom_left or top_right
variable
$bl_lon = $coords[0];
$bl_lat = $coords[1];
$tr_lon = $coords[2];
$tr_lat = $coords[3];

// calculate the approx center of the view -- note that this is innaccurate if the
user is not looking straight down
$userlon = (($coords[2] - $coords[0])/2) + $coords[0];
$userlat = (($coords[3] - $coords[1])/2) + $coords[1];

$response = '<?xml version="1.0" encoding="UTF-8"?>';
$response .= '<kml xmlns="http://earth.google.com/kml/2.0">';
$response .= '<Placemark>';
$response .= '<name>'.$timesnap.'</name>';
$response .= '<Point>';
$response .= "<coordinates>$userlon,$userlat,0</coordinates>";
$response .= '</Point>';
$response .= '</Placemark>';
$response .= '</kml>';
# set $myKMLCode together as a string
 $downloadfile="myKml.kml"; # give a name to appear at the client
 header("Content-disposition: attachment; filename=$downloadfile");
 header("Content-Type: application/vnd.google-earth.kml+xml; charset=utf8");
 header("Content-Transfer-Encoding: binary");
 header("Content-Length: ".strlen($response));
 header("Pragma: no-cache");
 header("Expires: 0");
echo $response;
?>
```

The code borrows heavily from a number of sources.[124]

This script puts into place a communication link between Google Earth and a server-side script. The next step is to substitute the KML that contains information about Flickr photos within the bounding box rather than the timestamp.

[124] http://bbs.keyhole.com/ubb/showflat.php?Number=279744 and
http://earth.google.com/kml/kml_tut.html#tracking_point and
http://www.fmepedia.com/index.php/Google_Earth_Data_Exchange_%28KML%29#NetworkLink

Google Mapplet that shows Flickr photos

Google has release Mapplets, a way of adding extensions to Goole Maps directly as little applications that run in a side panel. As of the time of writing, mapplets are in developer preview. You can access the mapplets at

`http://maps.google.com/preview`

Developer information is found at:

`http://www.google.com/apis/maps/documentation/mapplets/index.html`

You can find the source for a mapplet that allows users to search for Flickr pictures of a certain tag at:

`http://examples.mashupguide.net/ch13/helloworld.mapplet.1.xml`

The source is

```
<?xml version="1.0" encoding="UTF-8"?>
<Module>
<ModulePrefs title="Flickr photos"
                description="Show Flickr photos"
                author="Raymond Yee"
                author_email="raymondyee@mashupguide.net"
                height="150">
  <Require feature="sharedmap"/>
</ModulePrefs>
<Content type="html"><![CDATA[

<script>

  var map = new GMap2();

function genPhotoLink(photo) {

    var t_url = "http://farm" + photo.farm + ".static.flickr.com/" + photo.server +
"/" + photo.id + "_" + photo.secret + "_" + "t.jpg";
    var p_url = "http://www.flickr.com/photos/" + photo.owner + "/" + photo.id;

    return '<a href="' + p_url + '">' + '<img alt="'+ photo.title + '"src="' + t_url
+ '"/>' + '</a>';
}

// Creates a marker at the given point with the given msg.

function createMarker(point, msg) {
  var marker = new GMarker(point);
  GEvent.addListener(marker, "click", function() {
    marker.openInfoWindowHtml(msg);
  });
  return marker;
}

function createMarkerAndDiv (point,msg) {
    var marker, e, anchors, alink

    marker = createMarker(point, msg);
    e = document.createElement("div");

    e.innerHTML = msg +  "<a href='#'>Show</a><br>"
    anchors = e.getElementsByTagName('a')
    alink = anchors[anchors.length-1];
    alink.onclick = function(){marker.openInfoWindowHtml(msg);}

    return [marker,e]

}

function cb(s) {

    var rsp = eval('(' + s + ')');
    var marker, e

    // clear the photos
    map.clearOverlays();
    var pdiv = document.getElementById("pictures");
    pdiv.innerHTML = "Total number available is: " + rsp.photos.total + "<br/>";;

    // put the pictures on the map
    for (var i=0; i < rsp.photos.photo.length; i++) {
        var photo = rsp.photos.photo[i];

        var point = new GLatLng (photo.latitude, photo.longitude);
        var msg = photo.title + "<br>" + genPhotoLink(photo);

        [marker,e] = createMarkerAndDiv(point,msg);
        map.addOverlay(marker);
        pdiv.appendChild(e);


 //       var marker = createMarker(point, msg)
 //       map.addOverlay(marker);

 //       var e = document.createElement("div");
 //       e.innerHTML = msg +  "<a href='#'>Show</a><br>"
 //       var anchors = e.getElementsByTagName('a')
 //       var alink = anchors[anchors.length-1];

 //       alink.onclick = function(){marker.openInfoWindowHtml(msg);}
        //alink.onclick = function() {alert("hello"); return false;}
        //if (alink.captureEvents) alink.captureEvents(Event.CLICK);
//        pdiv.appendChild(e);
        //alert(alink.onclick);
    }
```

```
}

function get_pictures() {

    var API_KEY = "e81ef8102a5160154ef4662adcc9046b";
    fForm = document.getElementById('FlickrForm');

    map.getBoundsAsync(function(bounds) {
      var lat0 = bounds.getSouthWest().lat();
      var lon0 = bounds.getSouthWest().lng();
      var lat1 = bounds.getNorthEast().lat();
      var lon1 = bounds.getNorthEast().lng();



      var url = "http://api.flickr.com/services/rest/?method=flickr.photos.search" +
    "&api_key=" + API_KEY +
    "&bbox=" + lon0 + "%2C" + lat0 + "%2C" + lon1 + "%2C" + lat1  +
    "&per_page=" + fForm.per_page.value +
    "&page=" + fForm.page.value +
    "&format=json&nojsoncallback=1&extras=geo";

    var tagValue = fForm.tag.value;
    // search by tag only if the box is not blank.
    if (tagValue.length) {
      url = url + "&tags=" + fForm.tag.value;
    } else {
      url = url + "&min_upload_date=820483200";
    }

    _IG_FetchContent(url, cb);

      } //anonymous function

    ); //map.getBoundsAsync

} //get_pictures

</script>

<form action="#" onsubmit="get_pictures(); return false;" id="FlickrForm">
    <p>Search for photos with the following tag:
    <input type="text" size="20" name="tag" value="flower">
    at page <input type="text" size="4" name="page" value="1"> with
    <input type="text" size="3" name="per_page" value="10"> per page.
    <button type="submit">Go!</p>
</form>
<div id="pictures"><a href="#" onClick="alert('testing 1, 2, 3'); return
false;">Testing 1, 2,3</a></div>
```

```
]]></Content>
</Module>
```

Conclusions: Speculations on the Future

The online mapping arena is changing so quickly, and I obviously am not able to cover the details of all these changes. Nonetheless, it's helpful to speculate on what I believe to be the long-term trends in this area as a way of priming yourself for future changes.

I think that you'll see a migration of many features found in typical full-fledged GIS system -- e.g., shading of layers -- into programmable web applications. (e.g., [http://www.pushpin.com/pd_server.html Pushpin | Multi-layer map server])

Not surprisingly, we'll see the platform players (such as Google Maps) incorporate functionality started off as extensions to the platform into the platform itself. For example, sites such as mapbuilder.net provided a user interface for building Google (or Yahoo!) map before Google made it easier to build a Google Map via its "My Maps" functionality. Google's My Maps doesn't exactly duplicate mapbuilder.net but it's bound to win a major audience by virtue of its tight integration with Google Maps.

Google Mapplets are little applications that embedded in the Google Maps directly. (Note the contrast to the main Google Maps API allows Google Maps to be embedded in third-party websites) Again, because mapplets are located right in the context of http://maps.google.com, users can take advantage of increased functionality without having to go to a third-party website. Google Mapplets are to Google Maps as Google Gadgets are to iGoogle, Google Desktop, Google Page Creator. They are extension mechanisms for different part of the Google Platform.

We will see increased merging in 2D and 3D representations of the globe. Signs of such activity include

* Microsoft's integration of 3D and 2D views right in maps.live.com
* KML as a way of moving data between Google Earth and Google maps (Note the support for KML coming out of Yahoo! Pipes) ([http://blog.pipes.yahoo.com/2007/05/02/pipes-adds-interactive-yahoo-maps-kml-support-and-more/ Pipes Blog » Blog Archive » Pipes Adds Interactive Yahoo! Maps, KML Support (and More)])

To Learn More

Some references:

* *Building Google Maps Applications with PHP*
* *Building Google Maps Applications with Rails*
* *Google Maps Hacks : Tips & Tools for Geographic Searching and Remixing*
* *Hacking Google Maps and Google Earth*
* *Web Mapping Illustrated*
* *Mapping Hacks : Tips & Tools for Electronic Cartography*

* Google Mapki[125] -- the great wiki and source of information on Google Maps.

[125] http://mapki.com/wiki/Main_Page