Web Services and REST

Introduction

- Web sites are normally accessed by a browser guided by a person
- But we have seen that **programs** can also access a web site, return one or more pages, and scrape the site for information
- Web Services is the idea of offering the capabilities/information on a web site via a programming interface, so application programs can more readily access the information on the site
- *Web Services* are APIs for accessing a website's information across the Internet

Introduction (cont'd)

- The implementation of Web Services is roughly divided into three categories:
  - *Big Web Services* which involve XML messages that are communicated by the Simple Object Access Protocol (SOAP); the API is formally described using the Web Services Description Language (WSDL). These services are normally used for server to server communication, using additional protocols like XML Security and XML Encryption.
  - *REST* (Representational State Transfer) *Services* which use HTTP methods PUT, GET, POST and DELETE.
  - *Cloud Services* which provide cloud storage, application hosting, content delivery, and other hosting services.
- All three types of Web Services provide access through APIs.
- The rest of the slides will cover REST Services and Cloud Services

REST Services

- Many web sites are now offering their facilities through REST Web Services
- REST Services can be used to access sites that perform the following functions:
  - Web Search (e.g. Yahoo's BOSS Search, Google Custom Search
  - Geolocation (e.g. Yahoo's Placefinder)
  - Photo Sharing (e.g. Yahoo's Flickr, Google's Picasa [retired ☹]))
  - Social Networking (e.g. Facebook, Twitter, MySpace)
  - Mapping (e.g. Google Maps)

- Access is provided using one or both of these methods:
  - Direct URL, returning a response in one or more formats (XML, JSON, PHP)
  - Library-based APIs, embedded in JavaScript, Java, C#, Objective-C and other source and binary library formats
- Many of these services now require or include OAuth user authentication
  - Oauth is a standard for clients to access server resources on behalf of a resource owner
  - E.g. see http://en.wikipedia.org/wiki/OAuth
- Many of these services limit daily usage by a single website, and require payment when the thresholds are breached

Cloud Services

- Cloud Services covers a variety of hosting services:
  - Application Hosting (e.g. AWS, Google App Engine, FireHost, Microsoft Azure)
  - Backup and Storage (e.g. AWS)
  - Content Delivery (e.g. Netflix hosted by AWS)
  - E-commerce (Amazon.com e-commerce)
  - Media Hosting (e.g. Microsoft Azure, RackSpace, Streaming Media Hosting)
  - DNS Protection Services (e.g., CloudFlare)
  - Consumer Cloud Storage (e.g. Apple iCloud Drive, Dropbox, Microsoft OneDrive, Google Drive)

- Access is provided using one or both of these methods:
  - Dashboard
  - Library-based APIs, embedded in Java, C#, Objective-C and other binary library formats
- All these services are commercial services that require monthly payments
- The consumer cloud services provide limited, free basic storage

REST (Representational State Transfer)

- REST is a style of software architecture for distributed hypermedia systems (i.e. the Web)
  - Initially proposed by Roy Fielding in a 2000 doctoral dissertation
  - The World Wide Web is an example of REST
- There are three fundamental aspects of the REST Design Pattern
  - 1. client, 2. servers and 3. resources
    - Resources are typically represented as documents
    - Systems that follow Fielding's REST principles are often referred to as *RESTful*;

Resources

Every distinguishable entity is a resource

URLs

Every resource is uniquely identified by a URL

Simple Operations

(PUT,GET,POST,DELETE)

REST versus Other Approaches

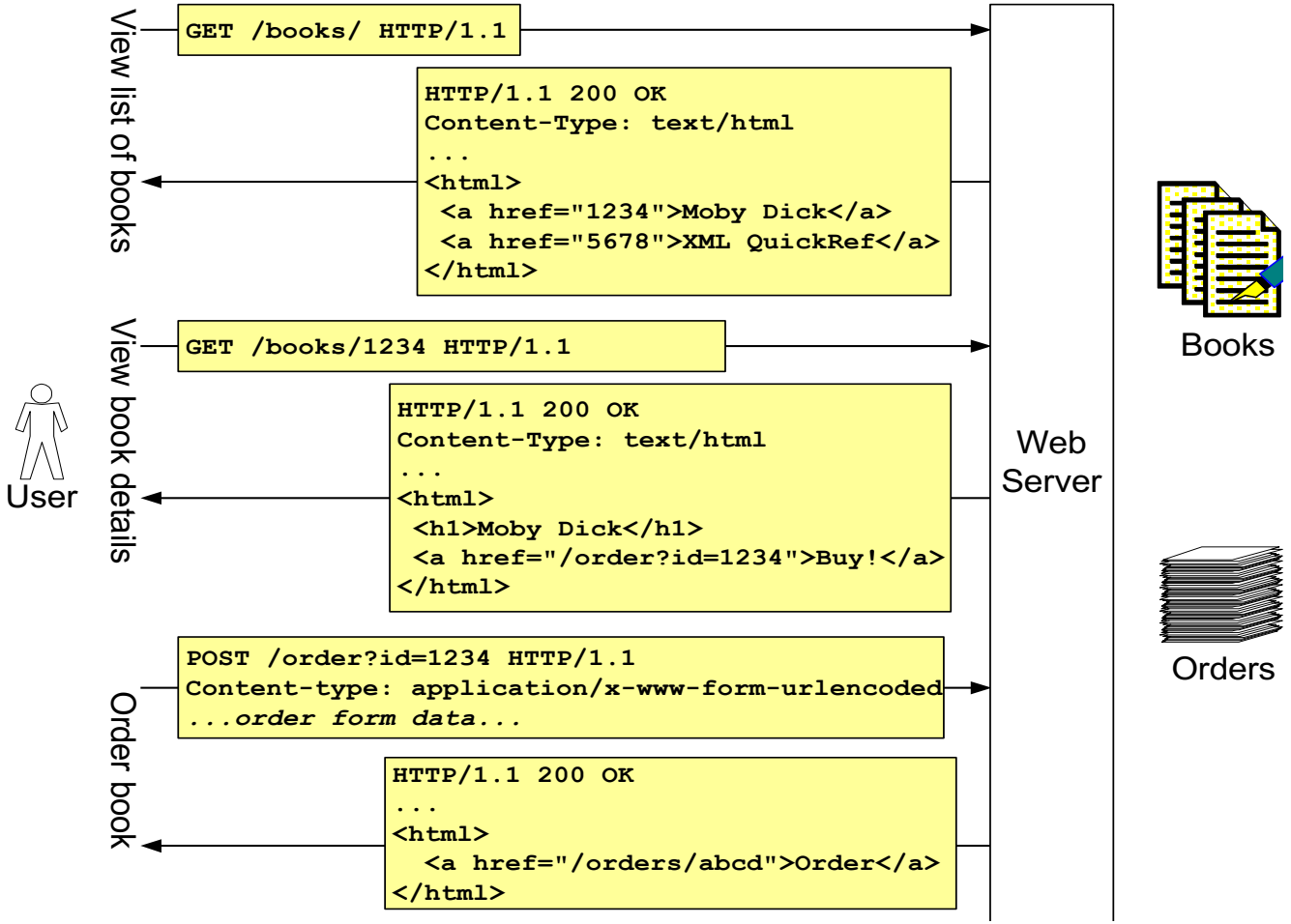
- REST
  - Software architectural style for distributed hypermedia systems like WWW
  - Quickly gained popularity through its simplicity
- SOAP
  - Protocol for exchanging XML-based message, normally using HTTP
  - Much more robust way to make requests, but more robust than most APIs need
  - More complicated to use
- XML-RPC
  - RPC protocol with XML as a encoding and HTTP as a transport
  - More complex than REST but much simpler than SOAP
- JSON-RPC
  - RPC protocol encoded in JSON instead of XML
  - Very simple protocol (and very similar to XML-RPC)

REST as Lightweight Web Services

- Much like Web Services, a REST service is:
  - Platform-independent (you don't care if the server is Unix, the client is a Mac, or anything else),
  - Language-independent (C# can talk to Java, etc.),
  - Standards-based (runs on top of HTTP), and
  - Can easily be used in the presence of firewalls.
- Like Web Services, REST offers no built-in security features, encryption, session management, QoS guarantees, etc. But also as with Web Services, these can be added by building on top of HTTP:
  - For security, username/password tokens are often used.
  - For encryption, REST can be used on top of HTTPS (secure sockets).
- One thing that is *not* part of a good REST design is cookies:
  - The "ST" in "RE**ST**" stands for "State Transfer", and indeed, in a good REST design operations are self-contained, and each request carries with it (transfers) all the information (state) that the server needs in order to complete it.

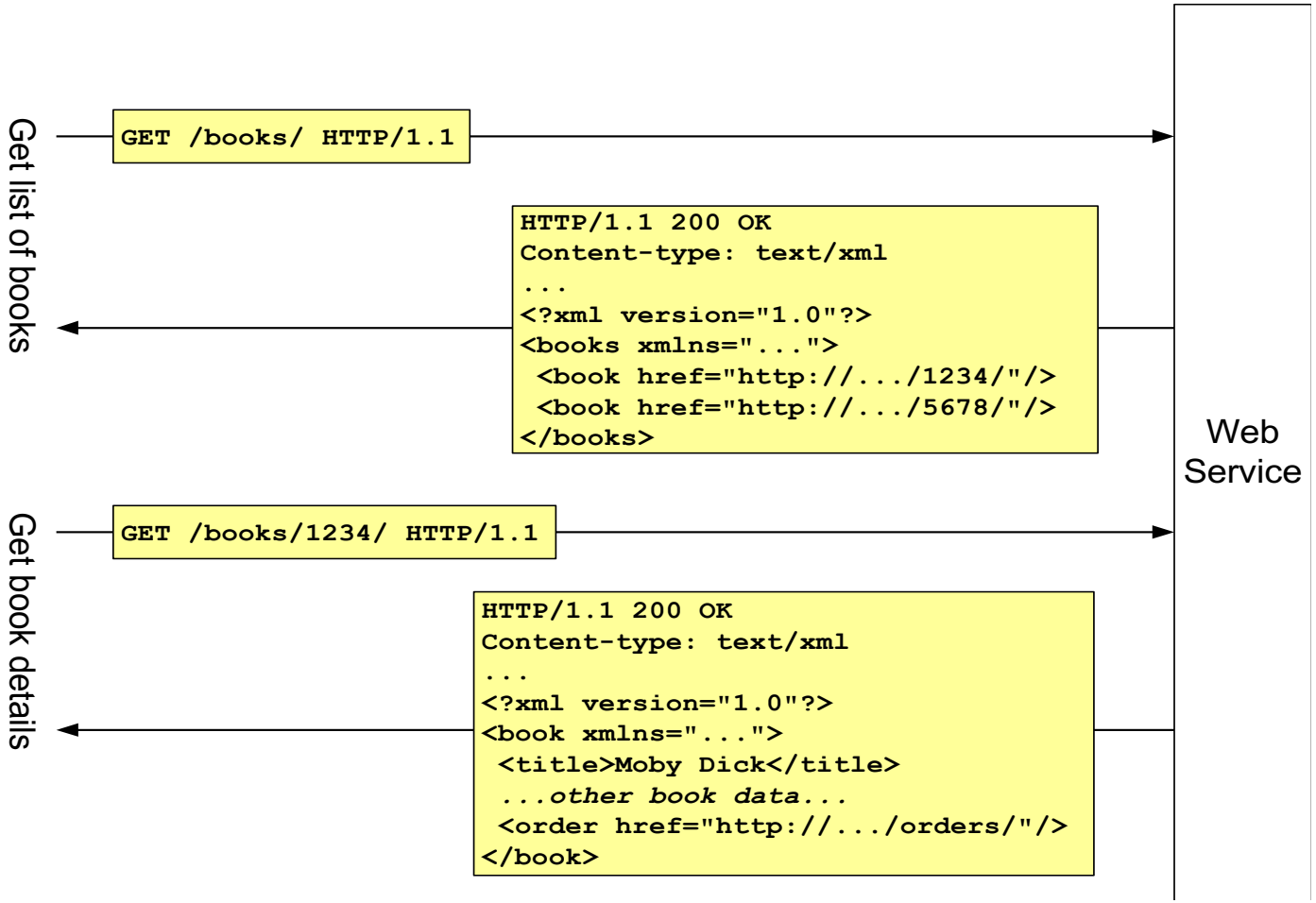
REST & the HTML Web

(Get book list, get book details, order book)

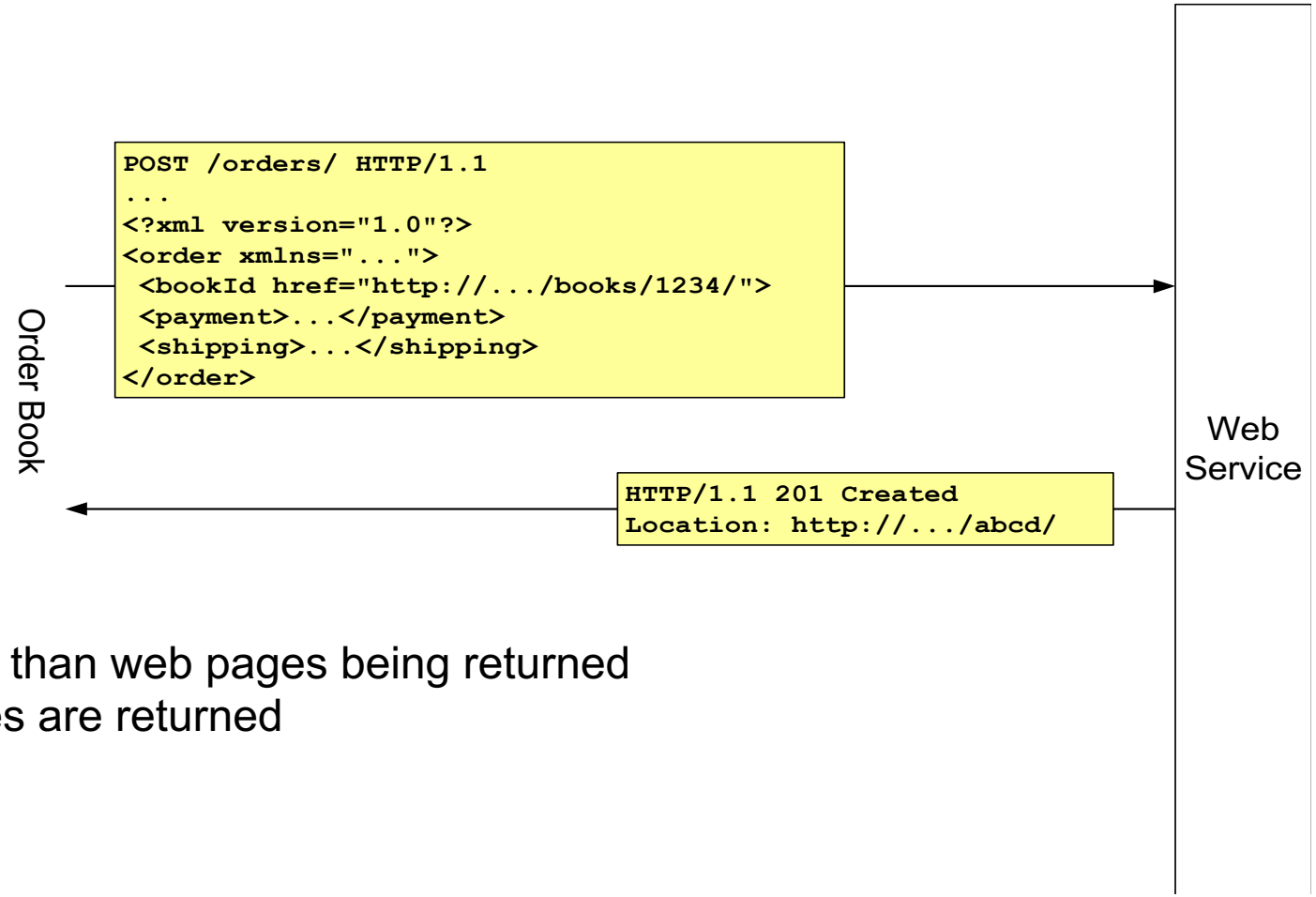


REST & the XML Web

(get book list, get book details)



REST & the XML Web (2)
(order book)



Rather than web pages being returned
xml files are returned

More Complex REST Requests

- REST can easily handle more complex requests, including multiple parameters.
- In most cases, you'll just use HTTP GET parameters in the URL.
- For example:

```
http://www.acme.com/phonebook/UserDetails?firstName=John&lastName=Doe
```

- If you need to pass long parameters, or binary ones, you'd normally use HTTP POST requests, and include the parameters in the POST body.
- As a rule,
  1. GET requests should be for read-only queries; they should not change the state of the server and its data.
  2. For creation, updating, and deleting data, use POST requests. (POST can also be used for read-only queries, as noted above, when complex parameters are required.
- While “legacy” REST services might use XML in their *responses* (as one way of organizing structured data), REST *requests* rarely use XML.
- Newer REST Services use JSON in their responses.
- As shown above, in most cases, request parameters are simple.
- One advantage of using XML is type safety. However, in a stateless system like REST, you should *always* verify the validity of your input, XML or otherwise!

REST Server Responses

- A server response in REST used to be an XML file; for example,

```
<parts-list>
  <part id="3322">
    <name>ACME Boomerang</name>
    <desc> Used by Coyote in <i>Zoom at the Top</i>, 1962 </desc>
    <price currency="usd" quantity="1">17.32</price>
    <uri>http://www.acme.com/parts/3322</uri> </part>
  <part id="783">
    <name>ACME Dehydrated Boulders</name>
    <desc> Used by Coyote in <i>Scrambled Aches</i>, 1957 </desc>
    <price currency="usd" quantity="pack">19.95</price>
    <uri>http://www.acme.com/parts/783</uri>
</part> </parts-list>
```

- However, other formats can also be used; REST is *not* bound to XML in any way. JSON is the response format recently used the most. Possible formats include CSV.
- One option that is *not* acceptable as a REST response format, except in very specific cases is HTML, or any other format which is meant for human consumption and is not easily processed by clients.
- The specific exception is, of course, when the REST service is documented to return a human-readable document; and when viewing the entire WWW as a RESTful application, we find that HTML is in fact the most common REST response format.

Yahoo Services Offered Via REST

- The Yahoo! Search Web Services are all REST services
- See http://developer.yahoo.com/everything.html for a list of all services; here are some specific ones
  - Flickr, https://www.flickr.com/services/api/
  - Yahoo Query Language (YQL), https://developer.yahoo.com/yql/
  - BOSS Search, https://developer.yahoo.com/boss/search/
    - Discontinued on March 31,2016
    - Using Yahoo's Search API you can build a customized search service
    - Priced at $1.80 / 1000 queries (was 80c a year ago)
    - Maximum of 50 results/query
    - Replaced by Yahoo Partner Ads (YPA), https://developer.yahoo.com/ypa/
  - BOSS Geo Services
    - Available for commercial and non-commercial use
    - https://developer.yahoo.com/boss/geo
    - PlaceFinder ($4/1000 queries) and PlaceSpotter ($8/1000 queries)

Flickr

- Photo-sharing community with APIs provide viewing and uploading access
- Request formats: REST, XML-RCP, SOAP
- Response Formats: REST, XML-RPC, SOAP, JSON, PHP. Supports JSONP.
- API Developer Kits available for 15 languages including ActionScript (Flash), Java (Android), .NET, Objective-C (iOS)
- Comprehensive number of API methods for authentication, blogs, contacts, favorites, galleries, people, photos
- Example Query:
  - https://api.flickr.com/services/rest/?method=flickr.photos.getRecent&api_key=f2cc26448280a762143ba4a865795ab4&format=json
  - (remove format parameter for XML results)
  - https required since June 2014

Flickr Sample JSON Result

jsonFlickrApi({"photos":{"page":1, "pages":10, "perpage":100, "total":1000, "photo":[{"id":"6879393174", "owner":"50010354@N05", "secret":"cf784500dd", "server":"7080", "farm":8, "title":"wjk_20110611_0092.jpg", "ispublic":1, "isfriend":0, "isfamily":0}, {"id":"6879393274", "owner":"31403543@N03", "secret":"af280ab218", "server":"6231", "farm":7, "title":"Imagen 415", "ispublic":1, "isfriend":0, "isfamily":0}, {"id":"6879393306", "owner":"66286618@N05", "secret":"7fc731bc3d", "server":"6237", "farm":7, "title":"IMG_6241-1", "ispublic":1, "isfriend":0, "isfamily":0}, {"id":"6879393338", "owner":"28935680@N03", "secret":"ec7444d9b6", "server":"7237", "farm":8, "title":"IMG_6756", "ispublic":1, "isfriend":0, "isfamily":0}, {"id":"6879393352", "owner":"32752988@N06", "secret":"be56f5751c", "server":"6046", "farm":7, "title":"AED_4586", "ispublic":1, "isfriend":0, "isfamily":0}, {"id":"6879393370", "owner":"29083790@N00", "secret":"ec89570135", "server":"6219", "farm":7, "title":"IMG_6546", "ispublic":1, "isfriend":0, "isfamily":0}, {"id":"6879393402", "owner":"50702313@N08", "secret":"18ecdd7871", "server":"7191", "farm":8, "title":"Group A 3", "ispublic":1, "isfriend":0, "isfamily":0}, {"id":"6879393418", "owner":"8502118@N08", "secret":"082968f6a9", "server":"6220", "farm":7, "title":"Buff-necked Ibis (Theristicus caudatus)", "ispublic":1, "isfriend":0, "isfamily":0}, {"id":"6879393440", "owner":"51425572@N04", "secret":"bc5f816ffb", "server":"6219", "farm":7, "title":"P2115768", "ispublic":1, "isfriend":0, "isfamily":0}, […]})

Partial Flickr Sample JSON Result With Formatting

jsonFlickrApi({"photos":{"page":1, "pages":10, "perpage":100, "total":1000,
"photo":[
{"id":"6879682760", "owner":"8348059@N02", "secret":"1ac6c7e2c4", "server":"6220", "farm":7,
"title":"DSC_0619", "ispublic":1, "isfriend":0, "isfamily":0},
{"id":"6879682762", "owner":"35772789@N02", "secret":"db5dffb91d", "server":"6117", "farm":7,
"title":"Dianna Romo 5", "ispublic":1, "isfriend":0, "isfamily":0},
{"id":"6879682776", "owner":"8091633@N05", "secret":"302174b53e", "server":"6118", "farm":7,
"title":"DSC_4259", "ispublic":1, "isfriend":0, "isfamily":0},
{"id":"6879682778", "owner":"58641881@N08", "secret":"c028082788", "server":"7212", "farm":8,
"title":"DSC_0777", "ispublic":1, "isfriend":0, "isfamily":0},
{"id":"6879682790", "owner":"32045507@N06", "secret":"d80d372bd2", "server":"6093", "farm":7,
"title":"IMG_9136", "ispublic":1, "isfriend":0, "isfamily":0},
{"id":"6879682792", "owner":"76919580@N08", "secret":"57e8d1cf8d", "server":"7277", "farm":8,
"title":"DSC01410", "ispublic":1, "isfriend":0, "isfamily":0},
{"id":"6879682796", "owner":"50838701@N04", "secret":"a3431e27e9", "server":"6042", "farm":7,
"title":"eP3274587", "ispublic":1, "isfriend":0, "isfamily":0},

YQL Web Service

- Yahoo Query Language is an expressive SQL-like language that lets you query, filter, and join data across Web services.
- With YQL, developers can access and shape data across the Internet through one simple language, eliminating the need to learn how to call different APIs.
- YQL exposes an SQL-like SELECT syntax. Through the SHOW and DESC commands, developers can discover the available data sources and structure without opening another Web browser. Open Data Tables enable developers to add tables for any data on the Web to our stable of API-specific tables. See:
  - http://www.datatables.org
- You can use YQL to write and modify data on Web services and applications using SQL keywords: INSERT, UPDATE, and DELETE. Tappid, query, number of results, radius, location (street, city, zip). See complete list at:
  - http://developer.yahoo.com/yql/guide/index.html
- Output formats: XML, JSON, JSONP (callback). See:
  - http://developer.yahoo.com/yql/guide/response.html
- YQL can be used for commercial purposes, with Yahoo approval.
- Per application limit (identified by your API Key): 100,000 requests per day
- Per IP limits: /v1/public/*: 2,000 calls per hour; /v1/yql/*: 20,000 requests per hour
  - https://developer.yahoo.com/yql/guide/overview.html#usage-info

YQL Query to access Zillow - Sample JSONP Result

http://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20**zillow.search**%20where%20address%20%3D%20%221835%2073rd%20Ave%20NE%22%20and%20citystatezip%20%3D%20%2298039%22%20and%20zwsid%20%3D%20%22X1-ZWz1cse68iatcb_13bwv%22&**format=json**&env=http%3A%2F%2Fdatatables.org%2Falltables.env&**callback=GetProperties**

GetProperties({"query":{"count":1,"created":"2014-10-31T18:01:31Z","lang":"en-us","results":{"searchresults":{"SearchResults":"http://www.zillow.com/static/xsd/SearchResults.xsd","xsi":"http://www.w3.org/2001/XMLSchema-instance","schemaLocation":"http://www.zillow.com/static/xsd/SearchResults.xsd http://www.zillowstatic.com/vstatic/6eb1265/static/xsd/SearchResults.xsd","request":{"address":"1835 73rd Ave NE","citystatezip":"98039"},"message":{"text":"Request successfully processed","code":"0"},"response":{"results":{"result":{"zpid":"49118839","links":{"homedetails":"http://www.zillow.com/homedetails/1835-73rd-Ave-NE-Medina-WA-98039/49118839_zpid/","graphsanddata":"http://www.zillow.com/homedetails/1835-73rd-Ave-NE-Medina-WA-98039/49118839_zpid/#charts-and-data","mapthishome":"http://www.zillow.com/homes/49118839_zpid/","comparables":"http://www.zillow.com/homes/comps/49118839_zpid/"},"address":{"street":"1835 73rd Ave NE","zipcode":"98039","city":"Medina","state":"WA","latitude":"47.627477","longitude":"-122.241942"},"zestimate":{"amount":{"currency":"USD","content":"128440851"},"last-updated":"10/28/2014","oneWeekChange":{"deprecated":"true"},"valueChange":{"currency":"USD","duration":"30","content":"-29689987"},"valuationRange":{"low":{"currency":"USD","content":"57798387"},"high":{"currency":"USD","content":"219633851"}},"percentile":"0"},"localRealEstate":{"region":{"id":"12669","name":"Medina","type":"city","zindexValue":"1,791,100","links":{"overview":"http://www.zillow.com/local-info/WA-Medina/r_12669/","forSaleByOwner":"http://www.zillow.com/medina-wa/fsbo/","forSale":"http://www.zillow.com/medina-wa/"}}}}}}}}}});

YQL Query to access Craigslist - Sample JSON Result

http://query.yahooapis.com/v1/public/yql?q=select%20*%20from%0A**craigslist.search**%20where%20location%3D%22sfbay%22%20and%20type%3D%22sss%22%20and%20query%3D%22schwinn%0Amountain%20bike%22&**format=json**&env=store%3A%2F%2Fdatatables.org%2Falltableswithkeys&**callback=**

{ "query": { "count": 1, "created": "2013-11-03T23:59:20Z", "lang": "en-US", "results": { "RDF": { "xmlns": "http://purl.org/rss/1.0/", "admin": "http://webns.net/mvcb/", "content": "http://purl.org/rss/1.0/modules/content/","dc": "http://purl.org/dc/elements/1.1/", "dcterms": "http://purl.org/dc/terms/“,”ev": "http://purl.org/rss/1.0/modules/event/", "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#", "syn": "http://purl.org/rss/1.0/modules/syndication/", "taxo": "http://purl.org/rss/1.0/modules/taxonomy/", "channel": { "about": "http://sfbay.craigslist.org/search/sss?format=rss&query=schwinn%0Amountain%20bike", "title": ["craigslist SF bay area | all for sale / wanted search \"schwinn\nmountain bike\"", "craigslist SF bay area | all for sale / wanted search \"schwinn\nmountain bike\""], "link": "http://sfbay.craigslist.org/search/sss?query=schwinn%0Amountain%20bike", "description": null,"language": "en-us","rights": "© 2013 craigslist","publisher": "robot@craigslist.org", "creator": "robot@craigslist.org", "source": "http://sfbay.craigslist.org/search/sss?format=rss&query=schwinn%0Amountain%20bike","type": "Collection", "updateBase": "2013-11-03T15:58:19-08:00","updateFrequency": "1","updatePeriod": "hourly", "items": { "Seq": { "li": [{ "resource": "http://sfbay.craigslist.org/scz/bik/4169456669.html" },.. { "resource": "http://sfbay.craigslist.org/sby/bik/4113012353.html" }] } } }, "item": [{ "about": "http://sfbay.craigslist.org/scz/bik/4169456669.html", "title": ["SCHWINN HOMEGROWN Mountain Bike, Made in USA!! (santa cruz) $500", "SCHWINN HOMEGROWN Mountain Bike, Made in USA!! (santa cruz) $500"], "link": "http://sfbay.craigslist.org/scz/bik/4169456669.html", "description": "1999 Schwinn Homegrown, 21 inch, X-Large, 26in wheels, 9-speed, XTR, XT and LX, RockShox Judy SL 80mm travel with Aluminum steerer. Was just professionally tuned up. All serviceable bearings were serviced with synthetic grease. The fork was overhaule [...]", "date": "2013-11-03T13:43:16-08:00", "language": "en-us", "rights": "© 2013 craigslist", "source": "http://sfbay.craigslist.org/scz/bik/4169456669.html", "type": "text", "issued": "2013-11-03T13:43:16-08:00" },..Issued": "2013-10-06T09:44:18-07:00" }] } } }}

Yahoo! BOSS Search API (retired)

- BOSS API is an updated service that provides RESTful access to Web, Image, News, and Spelling with a simple pricing scheme based on usage. The service also provides qualifying developer's access to Yahoo Search Advertising.
- See sample queries at:

https://developer.yahoo.com/boss/search/

- Search Web:

https://yboss.yahooapis.com/ysearch/web?q=iPod&sites=&format=json

- Search News:

https://yboss.yahooapis.com/ysearch/news?q=earthquake&sites=&format=json

- Also search spelling, related, and images.
- Request Parameters: See complete list at:

http://developer.yahoo.com/boss/geo/docs/location-parameters.html

– Output formats: XML, JSON

– Requires Oauth. See Oauth Authorization model at:

https://developer.yahoo.com/boss/search/boss_api_guide/oauth.html

- BOSS API specification at:

https://developer.yahoo.com/boss/search/boss_api_guide/api_spec.html

Yahoo! BOSS Geo Services - PlaceFinder

- Convert Lat/long to street address and vice-versa. Free Non-Commercial version available using YQL Tables: limited to 2000 queries per day per table per App ID.
- See Guide (for Free non-commercial use) at:

  https://developer.yahoo.com/boss/geo/docs/free_YQL.html#table_p
- REST Request URLs:

- Find coordinate of street address:

http://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20geo.placefinder%20where%20text%3D%22**<address>**%22&AppID=MyAppiD

- Find street address nearest to a point:

http://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20geo.placefinder%20where%20text%3D%**<latitude-longitude>**%22%20and%20gflags%3D%22R%22&AppID=MyAppID

- Request Parameters: See complete list at:

https://developer.yahoo.com/boss/geo/docs/location-parameters.html

– Output formats: XML, JSON, JSONP

- Sample request: see next slides
- Response fields defined at:

http://developer.yahoo.com/boss/geo/docs/supported_responses.html

Yahoo! PlaceFinder Sample JSON Result

http://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20geo.placefinder%20where%20text%3D%2225221600%2BPennsylvania%2BAvenue%2C%2BWashington%2C%2BDC%22&**format=json&callback=**

{ "query": { "count": 1, "created": "2013-11-03T23:16:16Z", "lang": "en-US", "results": { "Result":

{ "quality": "90", "latitude": "38.898788", "longitude": "-77.03894", "offsetlat": "38.898788", "offsetlon": "-77.03894", "radius": "400",

"name": "Pennsylvania Avenue", "line1": "Pennsylvania Avenue", "line2": "Washington, DC 20006", "line3": null, "line4": "United States",

"house": null, "street": null, "xstreet": null, "unittype": null, "unit": null, "postal": "20006", "neighborhood": null,

"city": "Washington", "county": "District of Columbia", "state": "District of Columbia", "country": "United States", "countrycode": "US", "statecode": "DC", "countycode": "DC", "uzip": "20006", "hash": null, "woeid": "23689633", "woetype": "20" } } }}

Yahoo! PlaceFinder Sample XML Result

http://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20geo.placefinder%20where%20text%3D%2225221600%2BPennsylvania%2BAvenue%2C%2BWashington%2C%2BDC%22

<?xml version="1.0" encoding="UTF-8"?>

<query xmlns:yahoo="http://www.yahooapis.com/v1/base.rng" yahoo:count="1" yahoo:created="2013-11-03T23:05:23Z" yahoo:lang="en-us">

<results><Result><quality>90</quality>

<latitude>38.898788</latitude><longitude>-77.03894</longitude>

<offsetlat>38.898788</offsetlat><offsetlon>-77.03894</offsetlon><radius>400</radius><name>Pennsylvania Avenue</name><line1>Pennsylvania Avenue</line1><line2>Washington, DC 20006</line2><line3/><line4>United States</line4><house/><street/><xstreet/><unittype/><unit/><postal>20006</postal><neighborhood/><city>Washington</city><county>District of Columbia</county><state>District of Columbia</state><country>United States</country><countrycode>US</countrycode><statecode>DC</statecode><countycode>DC</countycode><uzip>20006</uzip><hash/><woeid>23689633</woeid><woetype>20</woetype>

</Result>

</results>

</query><!-- total: 10 --><!-- engine4.yql.gq1.yahoo.com -->

Microsoft Bing Maps REST Services

- Bing Maps REST Services: http://msdn.microsoft.com/en-us/library/ff701713.aspx
- The Bing Spatial Data Services are REST-based services that offer three key functionalities: batch geocoding, point of interest (POI) data, and the ability to store and expose your spatial data.
- Used for performing tasks such as geocoding, reverse-geocoding, routing and static imagery.
- REST Request URLs:

- Find a location by Address:

http://dev.virtualearth.net/REST/v1/Locations/CA/adminDistrict/postalCode/locality/addressLine?includeNeighborhood=includeNeighborhood&maxResults=maxResults&key=BingMapsKey

- Find a location by query:

http://dev.virtualearth.net/REST/v1/Locations/1%20Microsoft%20Way%20Redmond%20WA%2098052?o=xml&key=BingMapsKey

- Find a location by point:

http://dev.virtualearth.net/REST/v1/Elevation/List?points=35.89431,-110.72522,35.89393,-110.72578,35.89374,-110.72606,35.89337,-110.72662&key=BingMapsKey

Microsoft REST Services (cont'd)

- Request Parameters: See complete list at:

http://msdn.microsoft.com/en-us/library/ff701715.aspx

– Output formats: XML, JSON (output=JSON), JSONP (jsonp=callback), and PHP

- Response fields (Location Data) defined at:

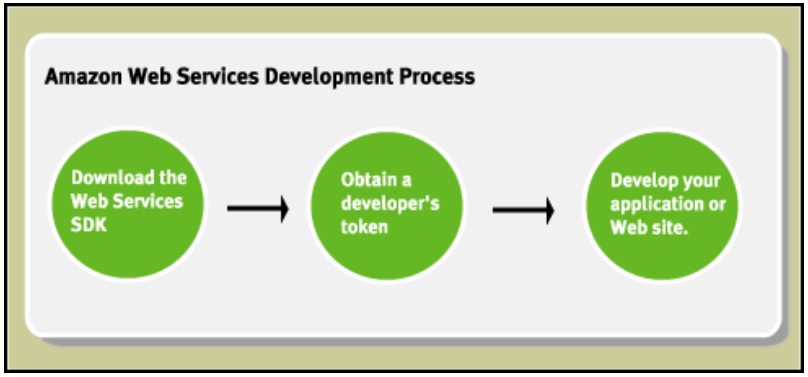
http://msdn.microsoft.com/en-us/library/ff701725.aspx

Amazon Now Offers Many Web Services

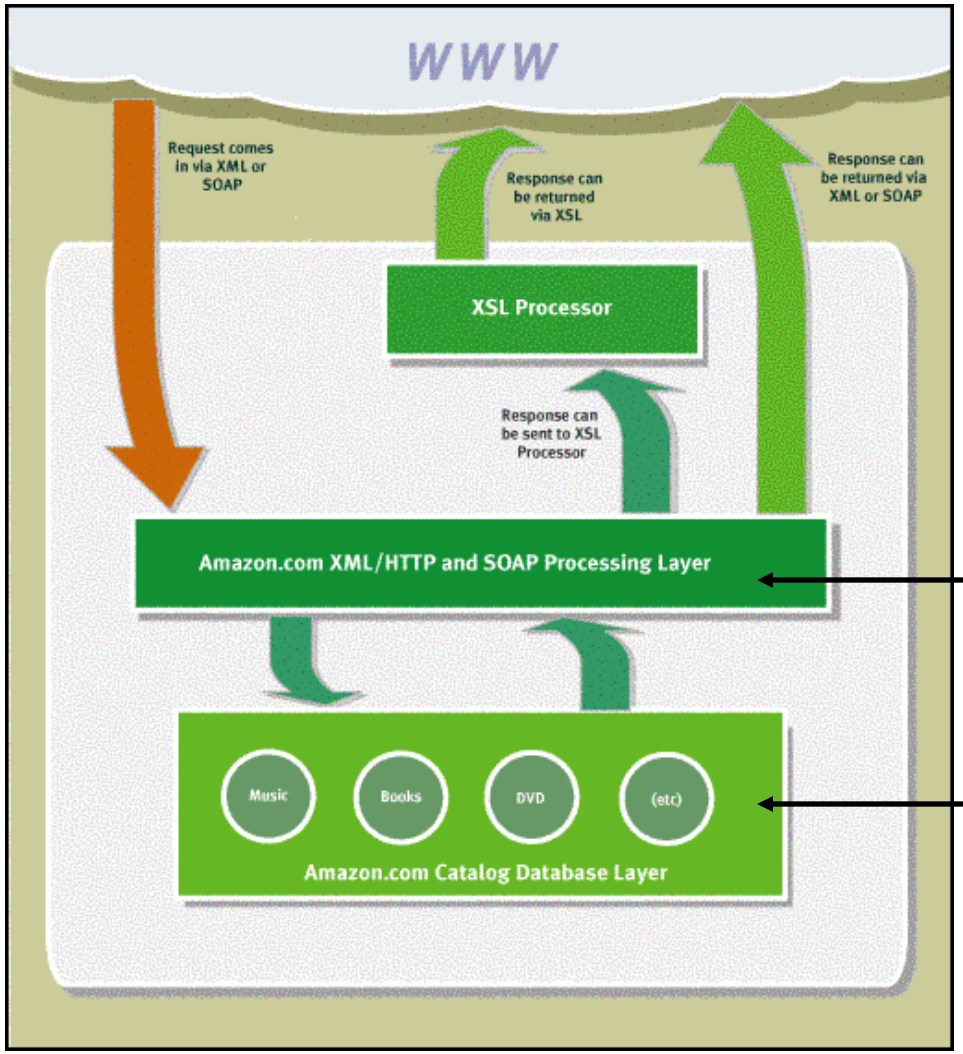
- Among them are
  - Amazon Associates Web Services (see next slides)
  - Amazon Elastic Compute Cloud (or EC2)
    - allows users to rent computers on which to run their own computer applications. EC2 allows scalable deployment of applications by providing a web service through which a user can boot an Amazon Machine Image to create a virtual machine, which Amazon calls an "instance", containing any software desired.
    - A user can create, launch, and terminate server instances as needed, paying by the hour for active servers, hence the term "elastic".
  - Amazon primarily charges customers in two ways:
    - Hourly charge per running virtual machine
    - Data transfer charge
  - As of November 2009, Amazon charges $0.084/hour ($61/month) for the smallest "On-Demand" virtual machine running Linux and twelve times that for the largest one running Windows.
  - "Reserved" instances can go as low as $31/month for a three-year prepaid plan. The data transfer charge ranges from $0.08 to $0.15 per gigabyte, depending on the direction and monthly volume.
  - See http://en.wikipedia.org/wiki/Amazon_Elastic_Compute_Cloud for details

Amazon Associates Web Services

- Amazon offers web services to 3 types of users:
  - **Associates**: third-party site owners wishing to build more effective sponsored affiliate links to Amazon products, thus increasing their referral fees
  - **Vendors**: sellers on the Amazon platform looking to manage inventory and receive batch product data feeds
  - **Developers**: third-party developers building Amazon-driven functionality into their applications
- Amazon Web Services provides software developers direct access to Amazon's technology platform and product data.
- Developers can build businesses by creating Web sites and Web applications that use Amazon products, charging and delivery mechanisms.
- Using Web services, you can now enable your Web site visitors to add products to Amazon.com shopping carts, wedding registries, baby registries and wish lists directly from your site.
- http://aws.amazon.com/



Amazon Associates Web Services Data Flow



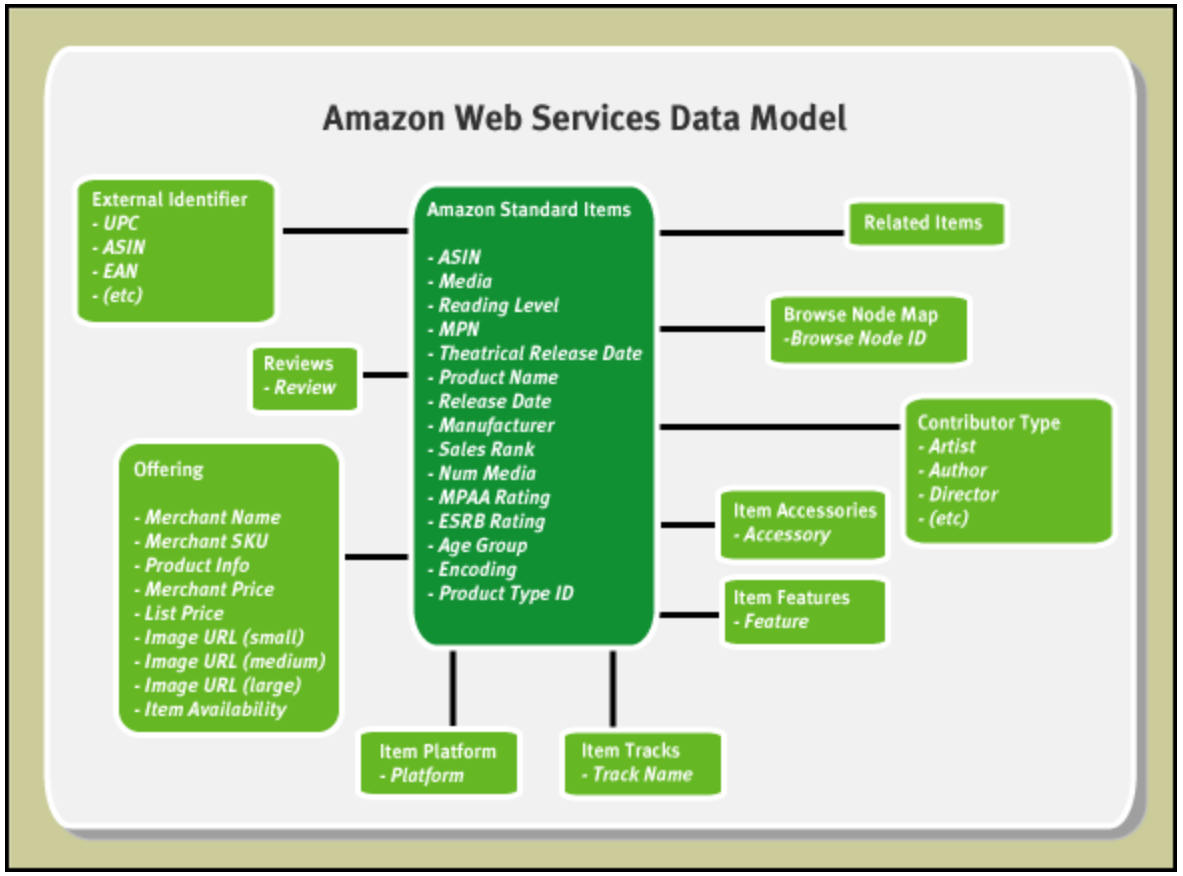
Amazon results can be returned either as XML files or as SOAP messages

XML/HTTP stands for REST

Data

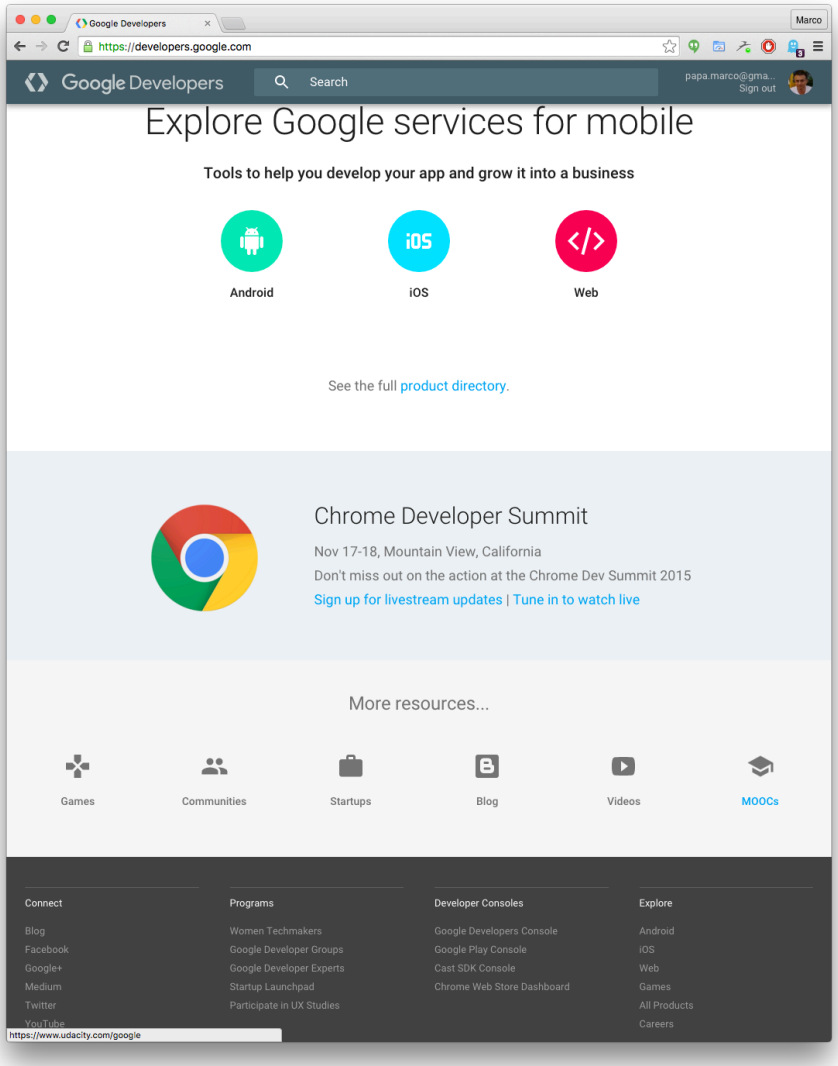
Amazon Web Services Data Model

Below is a graphical listing of the elements of the *A*mazon *W*eb *S*ervices data model.
The graphic represents the logical structure of AWS data.



Google APIs

- Available at:
  - http://developers.google.com
- APIs available for:
  - Google+
  - Android
  - App Engine
  - Chrome
  - Games
  - Google Maps
  - Google Apps
  - Google Play
  - Commerce
  - YouTube



Google App Engine

- Google App Engine lets you run web applications on Google's infrastructure.
  - There are no servers to maintain: You just upload your application (like AWS)
- You can serve your app from your own domain name or you can serve your app using a free name on the appspot.com domain.
  - You can limit access to members of your organization.
- Google App Engine supports apps written in several programming languages
  1. **Java** environment, including the JVM, and Java servlets.
  2. **PHP**
  3. **Python**. App Engine also features two dedicated **Python** runtime environments, each of which includes a fast Python interpreter and the Python standard library.
  4. **Go**. App Engine provides a **Go** runtime that runs natively compiled Go code.
  5. **Node.js** & **Ruby**. Included in the “flexible environment”.
- Download App Engine SDKs at:
  https://cloud.google.com/appengine/downloads
- You only pay for what you use and there are no set-up costs and no recurring fees
- Free daily limits are quite high (657K API calls, 200h connect time, 5GB storage)
- See: https://cloud.google.com/appengine/docs

Google Custom Search

- Enables searching over a website or a collection of websites
- Places a Google search box on a website that allows users to search the site
- Search results can be customized to match the design of the site
  - Google form to be filled out to create the custom search box

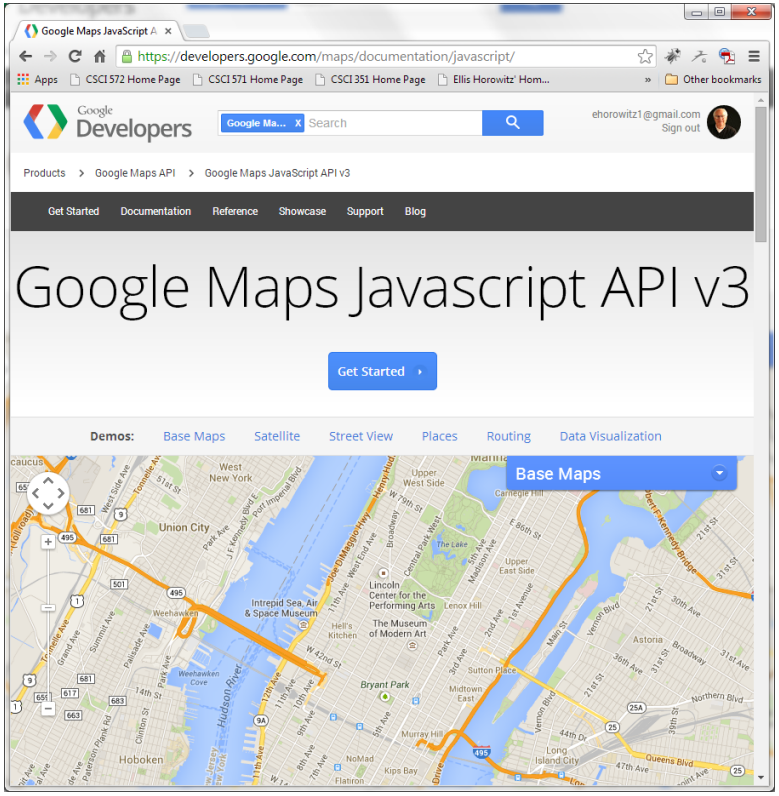
  http://www.google.com/cse/manage/create
- Create a search engine
  - Google Custom Search enables you to create a search engine for your website, your blog, or a collection of websites. You can configure your engine to search both web pages and images
- Search experience for users
  - Site search for your website
  - Topical search engine
  - Use structured data with Custom Search
- See: https://developers.google.com/custom-search/
- Documentation on implementing a “search box”:

https://developers.google.com/custom-search/docs/tutorial/implementingsearchbox

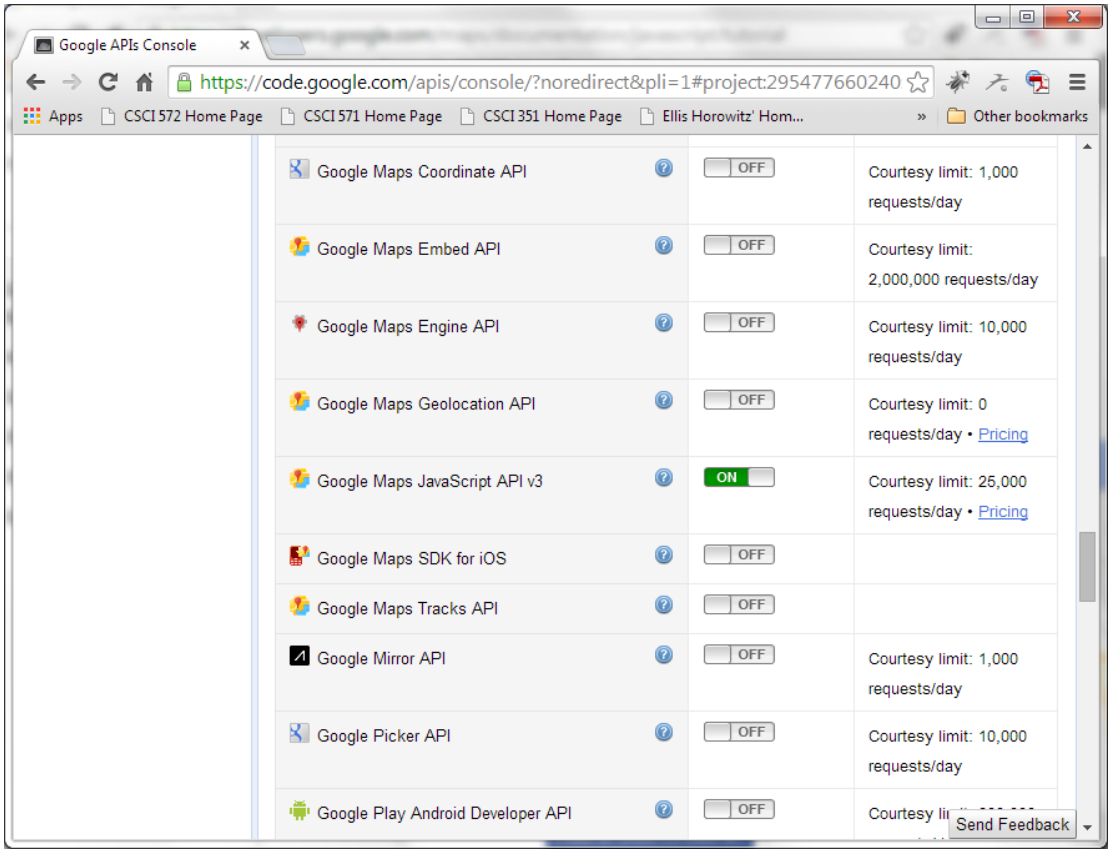
Google API Example: Google Maps API

- We will use the JavaScript API for Google maps
- https://developers.google.com/maps/documentation/javascript
- We will use their API, V. 3, click on "Get Started"
- First step: obtain an API key by selecting google Maps API v. 3
- Second step: return to `https://developers.google.com/maps/documentation/javascript/tutorial`

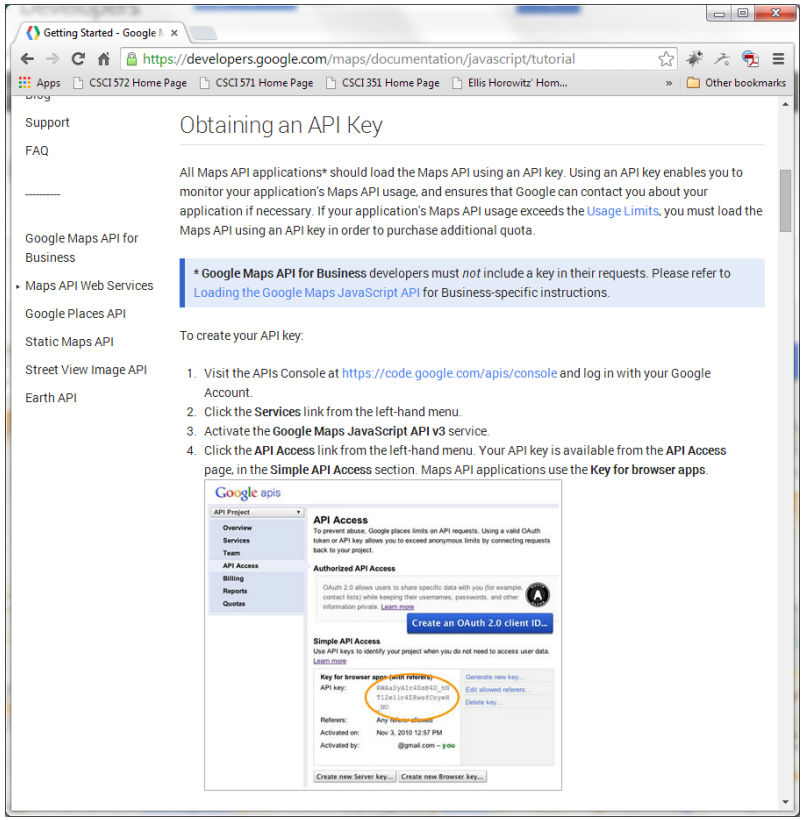
and examine the sample code



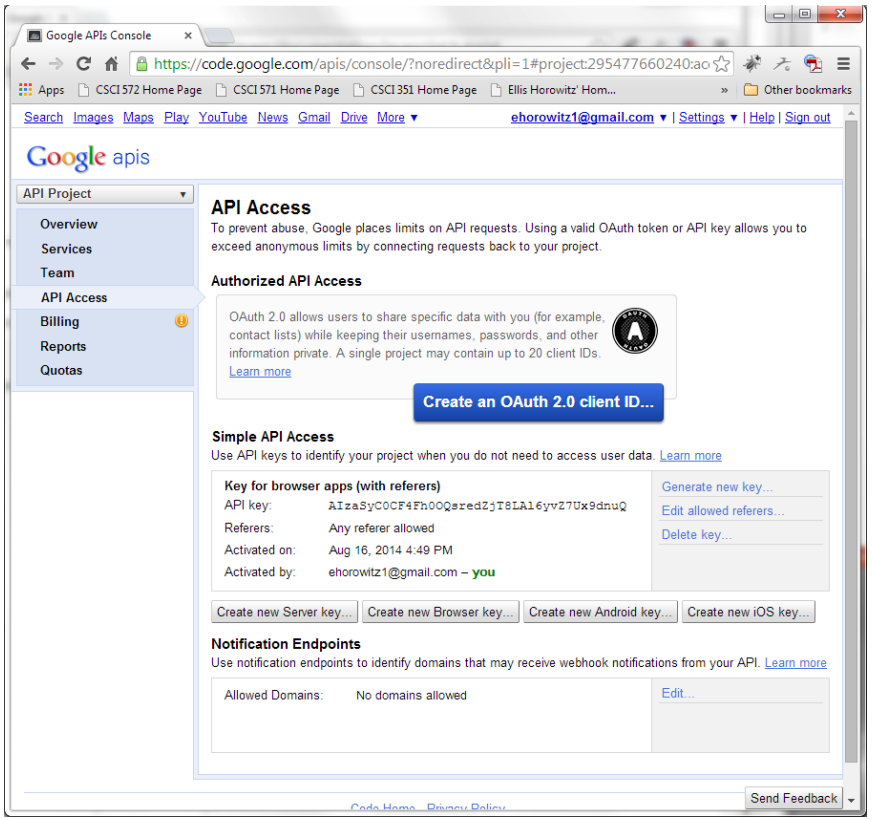
Activate Google Maps JavaScript API v3



Obtaining an API Key



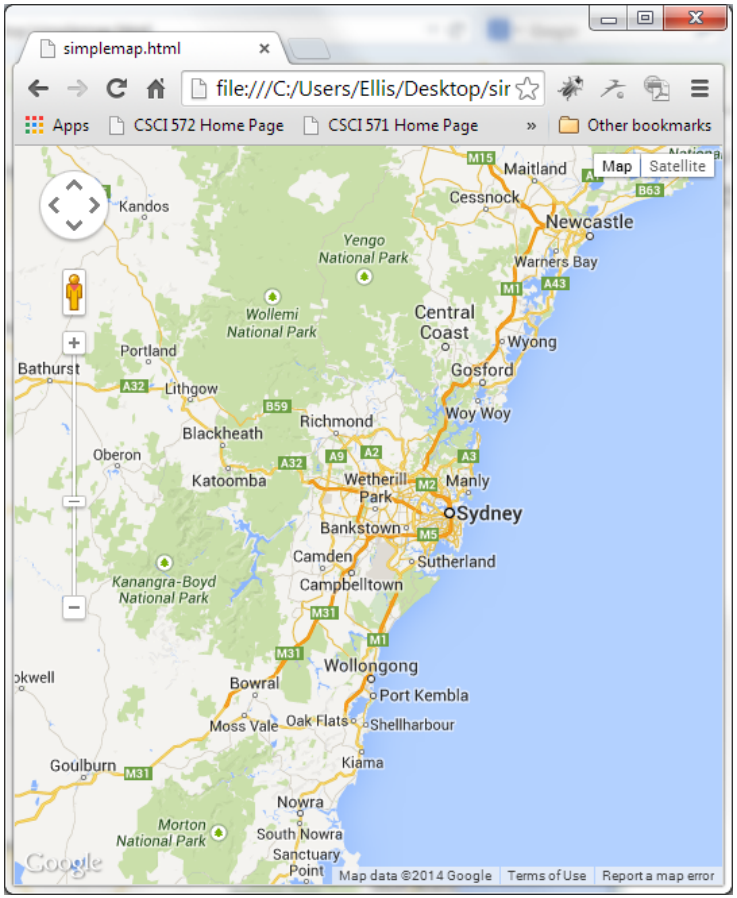
initial page



Returning a key result

Simple Maps Example

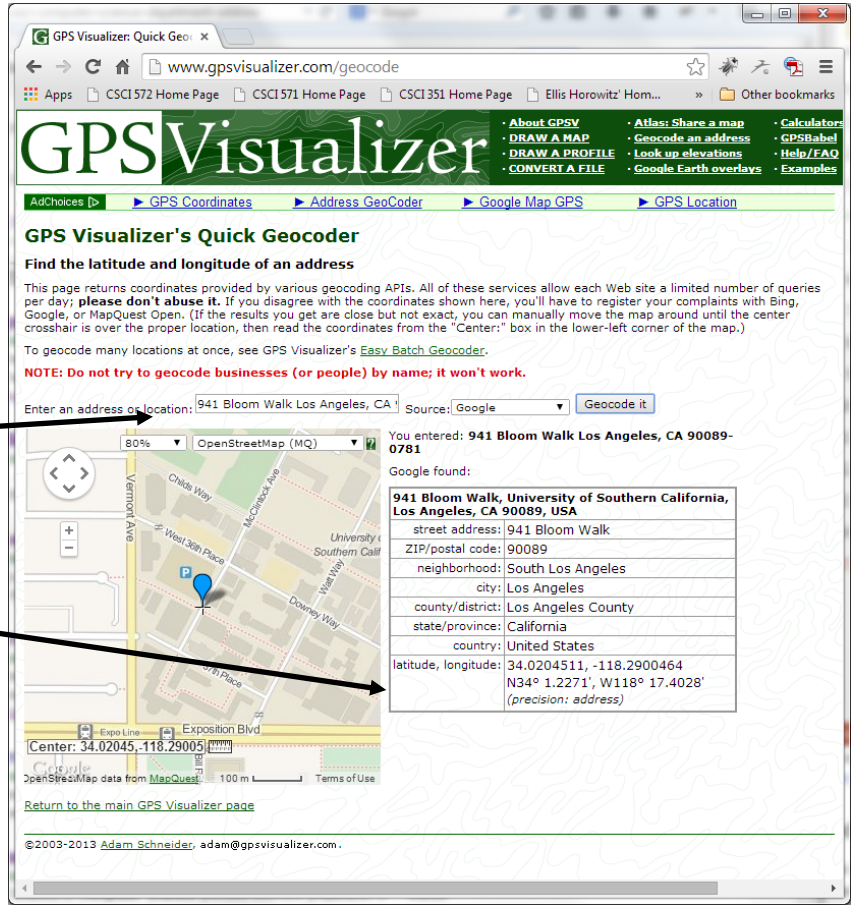
```
<!DOCTYPE html>
<html><head><meta name="viewport" content="initial-
scale=1.0, user-scalable=no" />
    <style type="text/css">
      html { height: 100% }
      body { height: 100%; margin: 0; padding: 0 }
      #map-canvas { height: 100% }
    </style>
    <script type="text/javascript"
src="https://maps.googleapis.com/maps/api/js?key=API
_KEY">
    </script>
    <script type="text/javascript">
      function initialize() {
        var mapOptions = {
          center: new google.maps.LatLng(-34.397,
150.644),
          zoom: 8   };
        var map = new
google.maps.Map(document.getElementById("map-
canvas"),
            mapOptions);    }
      google.maps.event.addDomListener(window,
'load', initialize);</script></head>
<body> <div id="map-canvas"/></body></html>
```



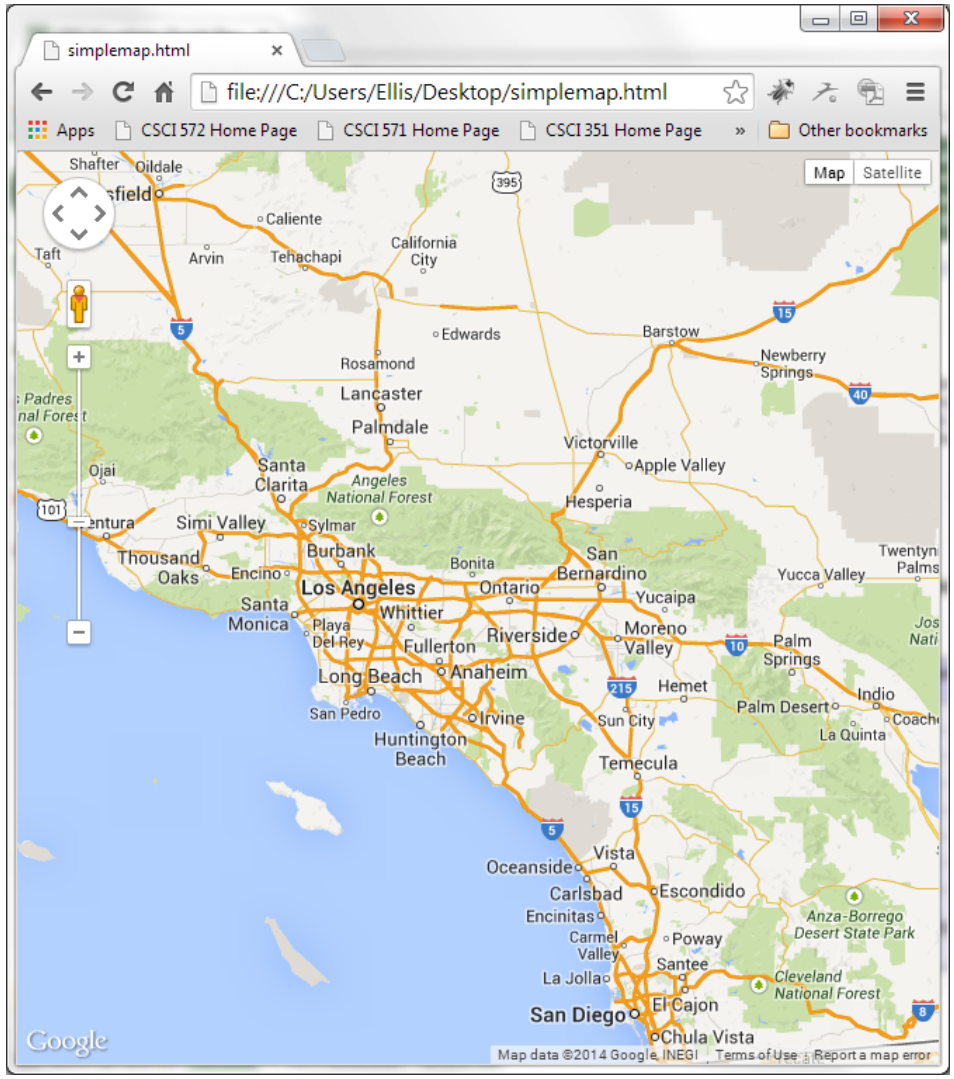
for many more details about this example see https://developers.google.com/maps/documentation/javascript/tutorial

Changing the Map's Center Point

- use geocoding to find the latitude/longitude of a local address
- let use a geocoding service, e.g. gpsvisualizer.com at http://www.gpsvisualize.com/geocode
- For an address we will use the CS dept
- the result is the lat/long

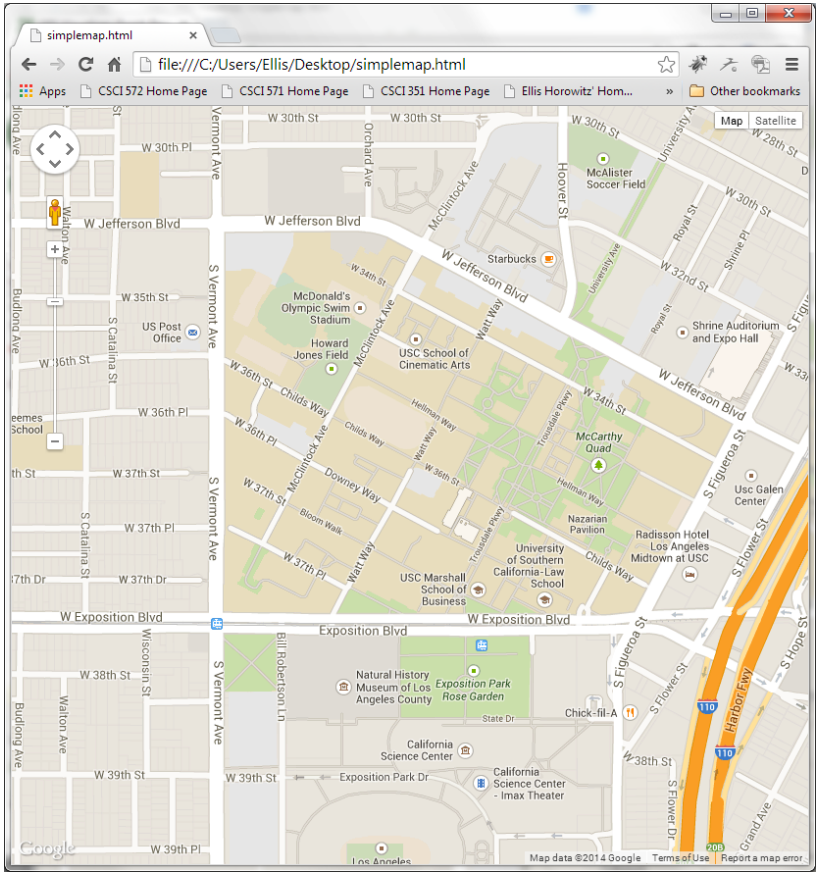


Simple Map with Lat/Long Change



Changing the Zoom Level

- the zoom level controls the distance above the map
- higher values cause the zoom to close in
- set the zoom value to 16 and the resulting map is produced



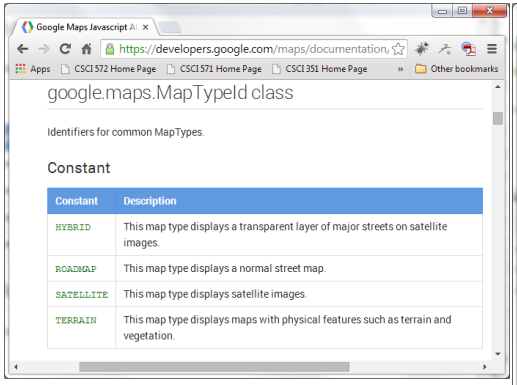
ROADMAP

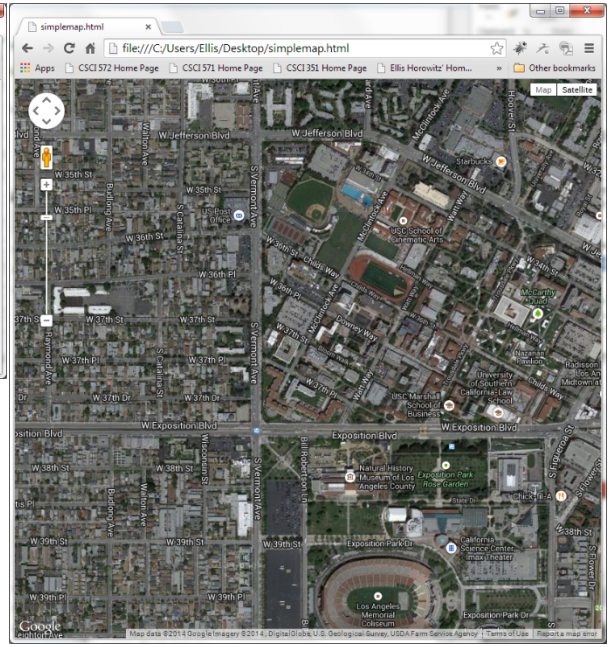
Adding a Marker to the Map

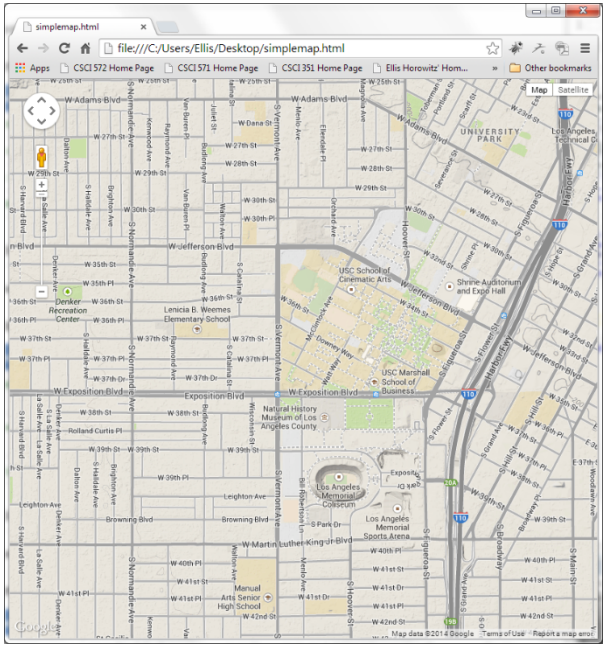
- But where is the CS dept.? we need to add a marker
- we see an example of a marker at https://developers.google.com/maps/documentation/javascript/examples/marker-simple

```
function initialize() {
  var myLatlng = new google.maps.LatLng(34.020, -118.290);
  var mapOptions = {  zoom: 4,  center: myLatlng  }
  var map = new google.maps.Map(document.getElementById('map-
canvas'), mapOptions);
  var marker = new google.maps.Marker({
      position: myLatlng,
      map: map,
      title: 'CS Dept'
  });
}
google.maps.event.addDomListener(window, 'load', initialize);
```

Change the Map Type





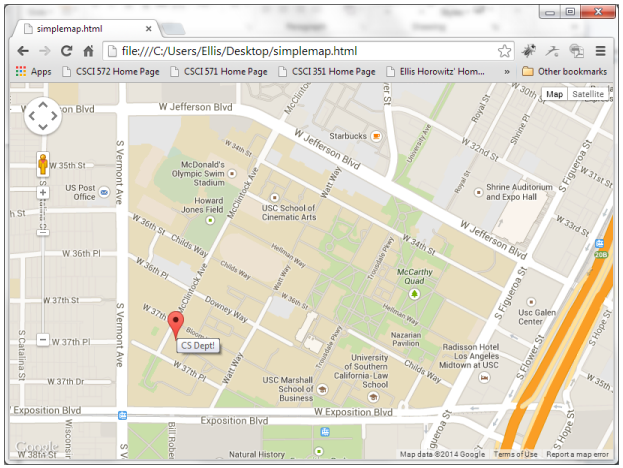


there are 4 map types:
HYBRID
ROADMAP
SATELLITE
TERRAIN

one can alter the map type by adding the line:
mapTypeId: google.maps.MapTypeId.HYBRID; or mapTypeId: google.maps.MapTypeId.TERRAIN

Add a Marker with Tool Tip

```
<!DOCTYPE html>

<html><head><meta name="viewport" content="initial-scale=1.0,
user-scalable=no" />

    <style type="text/css">

      html { height: 100% } body { height: 100%; margin: 0;
padding: 0 }

      #map-canvas { height: 100% }

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

src="https://maps.googleapis.com/maps/api/js?key=AIzaSyC0CF4Fh0OQs
redZjT8LAl6yvZ7Ux9dnuQ">

    </script>

    <script type="text/javascript">

function initialize() {

var mapOptions = {center: new google.maps.LatLng(34.020, -
118.290),

          zoom: 16   };

var map = new google.maps.Map(document.getElementById("map-
canvas"), mapOptions);

        var marker = new google.maps.Marker({

           position: new google.maps.LatLng(34.020, -118.290),

           map: map,

           title: 'CS Dept!'  })     }

  google.maps.event.addDomListener(window, 'load',
initialize);</script></head>

  <body> <div id="map-canvas"/></body></html>
```

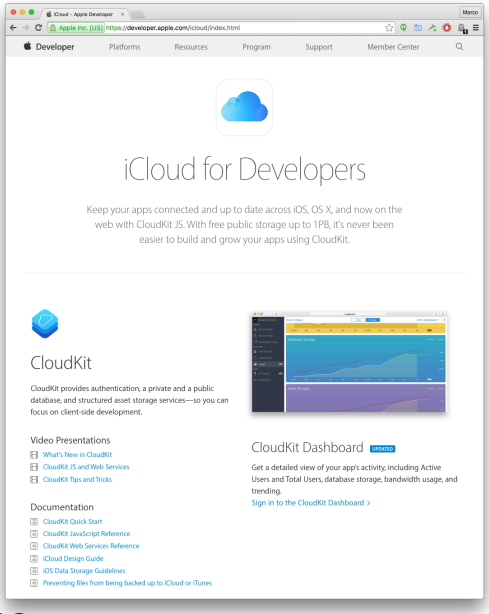


Adding a Popup Info Window to the Marker

```
<!DOCTYPE html><html><head><meta name="viewport" content="initial-
scale=1.0, user-scalable=no" />
    <style type="text/css">
html { height: 100% } body { height: 100%; margin: 0; padding: 0 }
#map-canvas { height: 100% }
    </style><script type="text/javascript"
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyC0CF4Fh0OQsredZjT8LA
l6yvZ7Ux9dnuQ">
    </script><script type="text/javascript">
function initialize() {
var mapOptions = {
center: new google.maps.LatLng(34.020, -118.290),zoom: 16   };
var map = new google.maps.Map(document.getElementById("map-canvas"),
mapOptions);
var marker = new google.maps.Marker({
           'position': new google.maps.LatLng(34.020, -118.290),
'map': map, 'title': 'CS Dept!'})}
 var contentString = '<div id="content">'+
 '<div id="siteNotice">CS Dept</div></div>';
var infowindow = new google.maps.InfoWindow({ content: contentString });
     google.maps.event.addDomListener(window, 'load', initialize);
     google.maps.event.addListener(marker, 'click', function() {
infowindow.open(map, marker) } );
</script></head><body> <div id="map-canvas"/></body></html>
```

Apple iCloud For Developers

- Apple's iCloud service places all information captured on any Apple device into the cloud, making it immediately available to all other Apple devices
- 5GB (free) – 50GB, 200GB, 1TB plans available at:
  - http://www.apple.com/icloud/
  - https://developer.apple.com/icloud/index.html
- iCloud APIs available for iOS 5 through 10 and OS X 10.9+
  - CloudKit framework
  - Storage API for Documents
  - Storage API for key-value data storage
  - Storage API for Core Data
  - Fallback Store (iOS 7+)
  - Account Changes (iOS 7+)
  - Manage iCloud Content (iOS 7+)
  - Xcode debugging (Xcode 5+)
  - iPhone simulator support (iOS 7+)



REST Best Practices

- 1. Provide a URI for each resource that you want exposed.
- 2. Prefer URIs that are logical over URIs that are physical. For example prefer

http://www.boeing.com/airplanes/747

Over:

http://www.boeing.com/airplanes/747.html

- Logical URIs allow the resource implementation to change without impacting client applications
- 3. As a corollary to (2) use nouns in the logical URI, not verbs. Resources are "things“ not "actions"
- 4. Make all HTTP GETs side-effect free.
- 5. Use links in your responses to requests. Doing so connects your response with other data. It enables client applications to be self-propelled. That is, the response itself contains info about "what's the next step to take".
- 6. Minimize the use of query strings. For example Prefer

http://www.parts-depot.com/parts/00345

Over

http://www.parts-depot.com/parts?part-id=00345

- 7. Use the slash "/" to represent a parent-child, whole-part relationship
- 8. Use a "gradual unfolding methodology" for exposing data to clients. That is, a resource representation should provide links to obtain more details.
- 9. Always implement a service using HTTP GET when the purpose of the service is to allow a client to retrieve a resource representation, i.e. don’t use HTTP POST