Google™ I/O 10

Google™ 10 I/O

Unleash Your Map Data

Cloud Computing for Geospatial Applications

Tom Manshreck

May 19, 2010

Live Blogging

View live notes and ask questions about this session on Google Wave:

http://bit.ly/a2zhnr

Google I/O 10

What Can You Do with Map Data?

- Private:
  - My Hikes
  - My Favorite Coffee Shops
  - Blog Entries by Location
- Shared:
  - Bike Trails
  - Restaurant Reviews
- Public:
  - Store Finders
  - Airline Maps
  - Crime Data

Why Store in the Cloud?

- Safer
- Exposed via Multiple Interfaces
  - My Maps UI
  - Maps Data API
- Accessible from any Browser/Application/Server
- Open Standards (Atom)

What is the Maps Data API?

- An Interface to Maps Data stored in the Google Cloud
- REST-ish Google Data API
  - Create, Read, Update, Delete
- Operations initiated through URL requests (Feeds)
  - HTTP POST, GET, PUT, DELETE
    - HTTP requests act as functions with parameters

Feeds of the Maps Data API

- Manipulates Three Entities:
  - Maps (collection of Data)
  - Features (datum per Map)
  - Access Control Lists (per Map)
- Provides Searches:
  - Spatial
  - Attribute Search

The Maps Data API

- Basic Maps Feed:
  - **`http://maps.google.com/maps/feeds/maps/`*`user@gmail.com`*`/full`**
  - – POST Acts to Create Map
  - – Initial Map Data can be provided as XML, CSV or KML
  - – Maps Created with the API are visible in My Maps
- Basic Feature Feed:
  - **`http://maps.google.com/maps/feeds/features/`*`user@gmail.com/mapID`*`/full/`**
  - – POST acts to create Feature in Map
  - – Features are KML:
    - **`<Point>`**
    - **`<LineString>`**
    - **`<Polygon>`**

The Maps Data API

- Authentication Options
  - AuthSub
  - OAuth
- Authorization via ACLs (Access Control Lists)
  - Specify Access on **`default`** (everyone) or per **`user@gmail.com`**:
  - Managed with Feeds too!

    `http://maps.google.com/maps/feeds/acl/maps/`*`user@gmail.com/mapId`*`/full`
  - Access is read for **`default`**, write for **`you@gmail.com`**

The Maps Data API

- URLs are a Pain!
- Use a Client Library Instead:
  - Java
  - Python
  - Javascript
- We'll use Javascript in this talk

Loading the Maps Data API: Javascript

- Use the Google Common Loader

```
<script type="text/javascript" src="http://www.google.com/jsapi"></script>

google.load('gdata', '2.x', {packages: ['maps']});
google.load('maps', '3', {other_params: ['sensor=true']});

google.setOnLoadCallback(initialize);
```

- Use AuthSub to Redirect to Google Login

```
var loginStatus;

function isLoggedIn() {
  loginStatus =  google.accounts.AuthSubStatus.getStatus();

  if (loginStatus == google.accounts.AuthSubStatus.LOGGED_OUT) {
    doLogin();
  } else if (loginStatus == google.accounts.AuthSubStatus.LOGGING_IN) {
    // Do nothing, as user is logging in
  }
}

function doLogin() {
  var token = google.accounts.user.login('http://maps.google.com/maps/feed');
}
```

- Have an IMG to Hold the Cookie Token

Example: Creating a Map

```
function addMap() {
  var mapFeedUrl = 'http://maps.google.com/maps/feeds/maps/default/owned';
  service.getMapFeed(mapFeedUrl, function(feedRoot){
    var newMap = new google.gdata.maps.MapEntry();
    newMap.setTitle(new google.gdata.atom.Text.create('Coffee Places'));
    newMap.setSummary(new google.gdata.atom.Text.create(
        'Coffee I've sampled around the world'));

    feedRoot.feed.insertEntry(newMap, function() {
        window.location.reload();
    }, errorHandler);
  }, errorHandler);
}
```

Application Design Specification

- Coffee Rating Mobile Application
- Use Maps Javascript API V3
- Uses Geolocation
- Allows You to Click and add Coffee Info
- Data Populated using Maps Data API
- Implement Search over Area for Coffee

The Maps API: Javascript

Populating a Map: Setting Up the Map

```
var initialLocation;
var map;
var geocoder;
var infowindow;
var service;
var markersArray = [];

var MAP_ID = 'user@gmail.com.mapID';

function initialize() {

  infowindow = new google.maps.InfoWindow();

  geocoder = new google.maps.Geocoder();

  var mapOptions = {
    zoom: 14,
    mapTypeId: google.maps.MapTypeId.ROADMAP;
  };

  map = new google.maps.Map(document.getElementById('map_canvas'), mapOptions);

  geolocate();

  google.maps.event.addListener(map, 'click', openCoffeeDialog);
}
```

The Maps API: Javascript

Populating a Map: Geolocation

```
function geolocate() {

  // Try W3C Geolocation
  if (navigator.geolocation) {
    navigator.geolocation.getCurrentPosition(function(position) {
      initialLocation = position.coords.latitude,position.coords.longitude);
      map.setCenter(initialLocation);
    }, function() {
      handleNoGeolocation();
    });
  } else {
    handleNoGeolocation();
  }
}

function handleNoGeolocation() {
  var moscone = new google.maps.LatLng(37.784182,-122.401509);
  initialLocation = moscone;
  map.setCenter(initialLocation);
}
```

The Maps API: Javascript

Populating a Map: Opening the Coffee Data Form

```
function openCoffeeDialog(event) {

  // Put login logic here for editing

  var location = new google.maps.LatLng();
  location = event.latLng;
  var lon = location.lng();
  var lat = location.lat();

  var html = "<form><table>" +
    "<tr><th colspan='2'>Coffee Rater 1.0</th></tr>" +
    "<tr><td>Name:</td><td><input type='text' size='16' id='name'/> </td> </tr>" +
    "<tr><td>Type of Coffee:</td>” +
      “<td><select id='type'>" +
        "<option value='coffee' SELECTED>filtered</option>" +
        "<option value='espresso'>espresso</option>" +
        "<option value='macchiatto'>macchiatto</option>" +
        "<option value='cappuccino'>cappuccino</option>" +
        "<option value='latte'>latte</option>" +
        "<option value='flatwhite'>flatwhite</option>" +
      "</select> </td></tr>" +
     "<tr><td>Rating:</td>” +
       “<td><select id='rating'>" +
         "<option value='5' SELECTED>*****</option>" +
         "<option value='4'>****</option>" +
         "<option value='3'>***</option>" +
         "<option value='2'>**</option>" +
         "<option value='1'>*</option>" +
       "</select> </td></tr>" +
```

The Maps API: Javascript

Populating a Map: Getting the Coffee Data

```
      "<tr><td></td><td>" +
      "<input type='hidden' id='lat' value='" + lat + "' />" +
      "<input type='hidden' id='lon' value='" + lon + "' />" +
      "<input type='button' value='Add Coffee' onclick='saveData(this.form)'/>" +
    "</td></tr></table></form>";

    infowindow.setContent(html);
    infowindow.setPosition(event.latLng)
    infowindow.open(map);
}

function saveData(form) {

  var establishment = form.name.value;
  var coffeeType = form.type.value;
  var coffeeRating = form.rating.value;
  var location = form.lon.value + ",” + form.lat.value;
  addCoffee(establishment,coffeeType,coffeeRating,location);
}
```

The Maps API: Javascript

Populating a Map: Getting the Coffee Data

To Add a Feature:

- Get the Map feed for your Map
- Get the **MapEntry** entity
- Get the **MapEntry**'s feature feed URL
- Populate a **FeatureEntry** entity
- Insert the **FeatureEntry** using the Feature Feed Url

```
function addCoffee(place,type,rating,location) {
  var mapUrl = "http://maps.google.com/feeds/maps/" + MAP_ID;

  service.getMapEntry(mapUrl, function(entryRoot) {
    var coffeeMap = entryRoot.entry;
    var featureFeedUrl = coffeeMap.getContent().getUri();

    addCoffeeFeature(featureFeedUrl,place,type,rating,location);
  }, errorHandler);
}
```

The Maps API: Javascript

Populating a Map: Getting the Coffee Data

```
function addCoffeeFeature(featureFeedUrl,place,type,rating,location) {
  service.getFeatureFeed(featureFeedUrl, function(feedRoot) {
    var newFeature = new google.gdata.maps.FeatureEntry();

    var title = new google.gdata.atom.Text();
    title.setText(place);
    newFeature.setTitle(title);

    newFeature.addCustomProperty({name: 'coffee', value: type});
    newFeature.addCustomProperty({name: 'rating', value: rating});

    var kmlContent =  new google.gdata.maps.KmlContent();
    var kmlString = '<Placemark><Point><coordinates>' +
      location +
    '</coordinates></Point></Placemark>';

    kmlContent.setText(kmlString);
    kmlContent.setType(
        google.gdata.maps.KmlContent.TYPE_APPLICATION_VND_GOOGLE_EARTH_KML_XML);

    newFeature.setContent(kmlContent);

    feedRoot.feed.insertEntry(newFeature, function() {window.location.reload();},
      errorHandler);
  }, errorHandler);
}
```

The Maps API: Javascript

Populating a Map: Getting the Coffee Data

Coffee Data Entry

The Maps API: Javascript

Searching Over Your Data

To Search for Features:

- Use a Feature Feed `/snippet?` query
- Pass Attribute parameters in a **mq** parameter as key/value pair
  - Attributes are passed as an array in [] brackets
  - Attributes must be exact match
  - Example: mq=[rating:5]
- Spatial searches are by **radius** or bounding **box**
- **radius** requires a **lat** and **lng** center
- **radius** is expressed in meters
- Optional **sortby** parameter returns results in specified order

- Sample query:

- ?mq=[rating:5][type:latte]&lat=37.1212111&lng=-112.545343&radius=1609.344&sortby=distance

The Maps Data API

Searching Over Your Data

```
function findCoffee() {

  var MAPS_FEED_URI = 'http://maps.google.com/maps/feeds/features' + MAP_ID +
      '/snippet?';

  var address = map.getCenter();

  var lat = address.lat();
  var lng = address.lng();
  var rating = document.getElementById('find_rating').value;
  var radius = document.getElementById('miles').value;
  radius *= 1609.344;

  // Create the attribute filters.
  var filters = new Array();
  filters.push('[rating:' + rating + ']');

  // Set up query parameters
  filters =  'mq=' + filters;
  radius = 'radius=' + radius;
  lat = 'lat=' + lat;
  lng = 'lng=' + lng;
```

(cont'd)

The Maps Data API

Searching Over Your Data

```
featureFeedUrl = MAPS_FEED_URL + lat + '&' + lng + '&' + filters + '&' + radius
    + '&sortby=distance';


service.getFeatureFeed(featureFeedUrl, function(feedRoot) {
  var feed = feedRoot.feed;
  var features = feed.getEntries();

  showResults(features);

}
```

The Maps Data API

Showing Your Results

```
function showResults(results) {

  var bounds = new google.maps.LatLngBounds();

  clearMarkers();

  for (var i = 0; i < results.length; i++) {
    var feature = results[i];
    var title = feature.getTitle.getText();
    var extendedData = feature.getCustomProperties();
    var coffeeType = extendedData[0].getValue();
    var rating = extendedData[1].getValue();

  // Get the KML Data
  var content = feature.getContent().getText();
  var parser = new DOMParser();
  var kml = parser.parseFromString(content, “text/xml”);
  var coordNode = kml.documentElement.getElementsByTagName(‘coordinates’):
  var coords = coordNode.item(0).firstChild.nodeValue.split(‘,’);

  var marker = new google.maps.Marker({
    position: new google.maps.LatLng(coords[1],coords[0]),
    title: title,
    map: map
  });

  (Cont’d)
```

The Maps Data API

Showing Your Results

```
      attachContent(marker, title, coffeeType, rating);

      bounds.extend(marker.position);

      markersArray.push(marker);

    }

  map.fitBounds(bounds);
}

function attachContent(marker,title,coffeeType,rating) {
  google.maps.event.addListener(marker, 'click', function() {
    infowindow.setContent("<b>" + title + "</b><br/>" + "type: " + coffeeType +
        "rating: " + rating);

    infowindow.setPosition(marker.position);
    infowindow.open(map,marker);
  });
}

function clearMarkers() {

  if (markersArray) {
    for (i in markersArray) { markersArray[i].setMap(null); };
    markersArray.length = 0;
  }
}
```

The Maps API: Javascript

Populating a Map: Getting the Coffee Data

Coffee Data Search

Google™ 10 I/O

A Case Study: IHG

Connecting our Customers with Data

Where are the IHG Hotels?

- 4,500+ Hotels
- Blanket the World with our Data
- Put our Data Closer to our Customers
- Focus on Business and not Infrastructure
- Apps Outside our Firewall have Easy Access

Connecting our Customers with Data

Where are the IHG Hotels?

- 4,500+ Hotels
- Blanket the World with our Data
- Put our Data Closer to our Customers
- Focus on Business and not Infrastructure
- Apps Outside our Firewall have Easy Access

Connecting our Customers with Data

Where are the IHG Hotels?

- Upload our Hotel Data Nightly
- Simple Rest Query to Perform Search

Sample Query #1
http://maps.google.com/maps/feeds/features/20266914295378485 6149/000485776b018b016c69b/snippet?lat=33.920716&lng=-84.338547&radius=50000&sortby=distance

Snippet of XML Results

<Point>
<coordinates>**-75.528681,40.576854,0.0**</coordinates>
</Point>
<gd:customProperty name="**hotelCode**">**ABEDP**</gd:customProperty>
<gd:customProperty name="**imageUrl1**">**http://www.ichotelsgroup.com/hotelmedia/repository/hotelimages/ABEDP/WELCM_EXTR_06_C.jpg**
</gd:customProperty>

Connecting our Customers with Data

Where are the IHG Hotels?

- Upload our Hotel Data Nightly
- Simple Rest Query to Perform Search

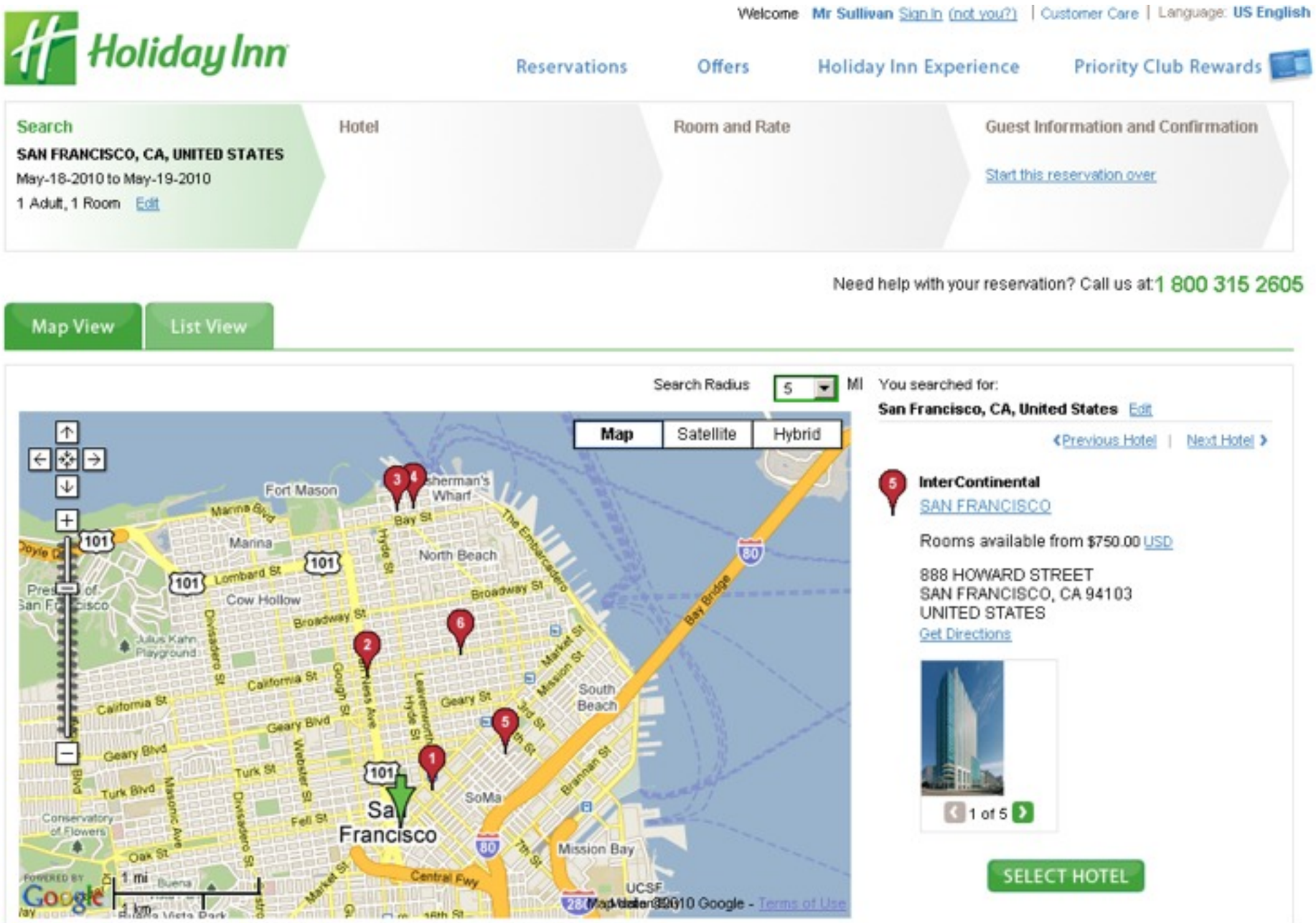
Sample Query #1
http://maps.google.com/maps/feeds/features/20266914295378485 6149/000485776b018b016c69b/snippet?lat=33.920716&lng=-84.338547&radius=50000&sortby=distance

Snippet of XML Results

```
-<Point>
  <coordinates>-75.528681,40.576854,0.0</coordinates>
  </Point>
  <gd:customProperty name="hotelCode">ABEDP</gd:customProperty>
  <gd:customProperty name="imageUrl1">http://www.ichotelsgroup.com/hotelmedia/repository/hotelimages/ABEDP/WELCM_EXTR_06_C.jpg
  </gd:customProperty>
```

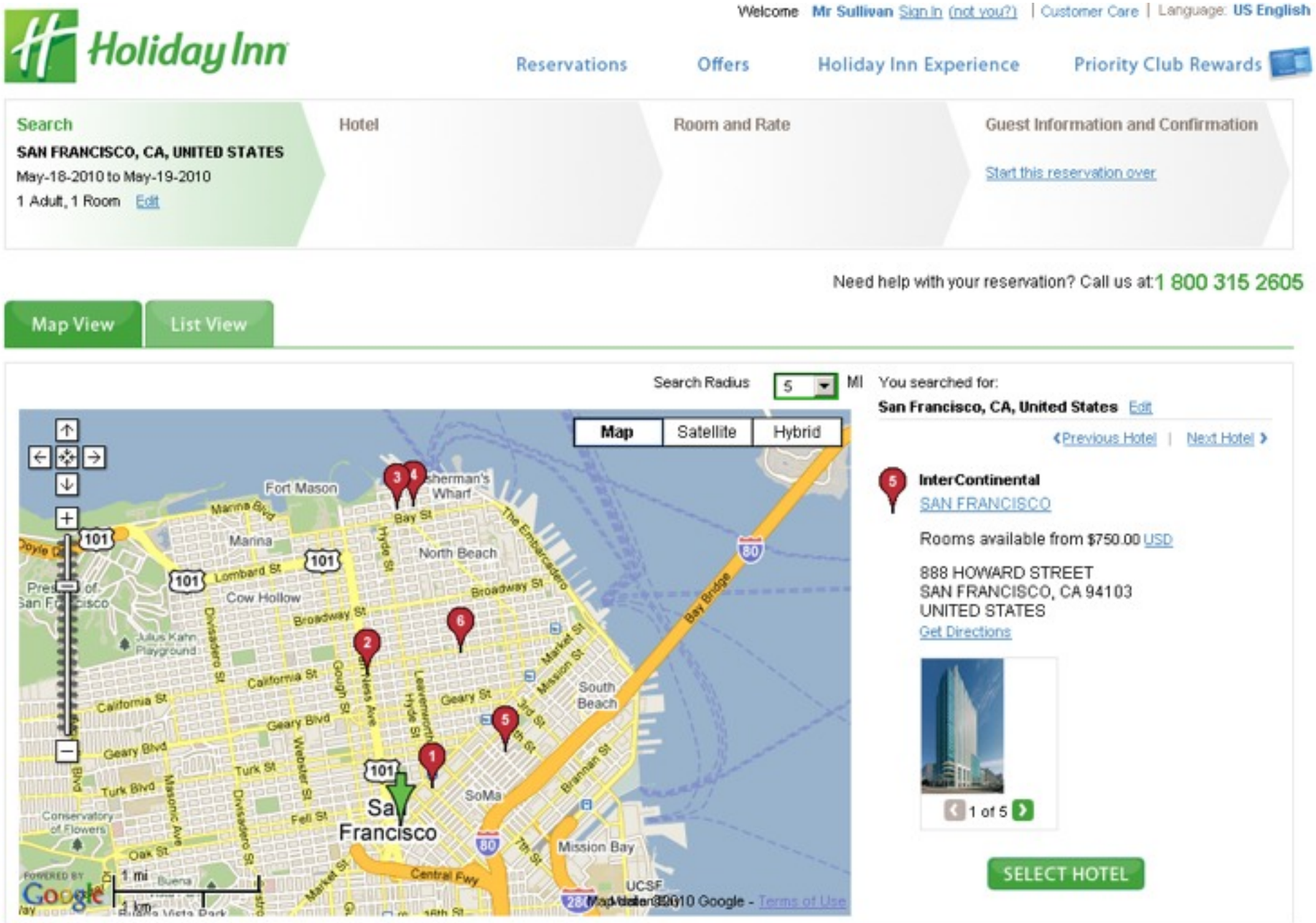
Connecting our Customers with Data

Where are the IHG Hotels?



Connecting our Customers with Data

Where are the IHG Hotels?



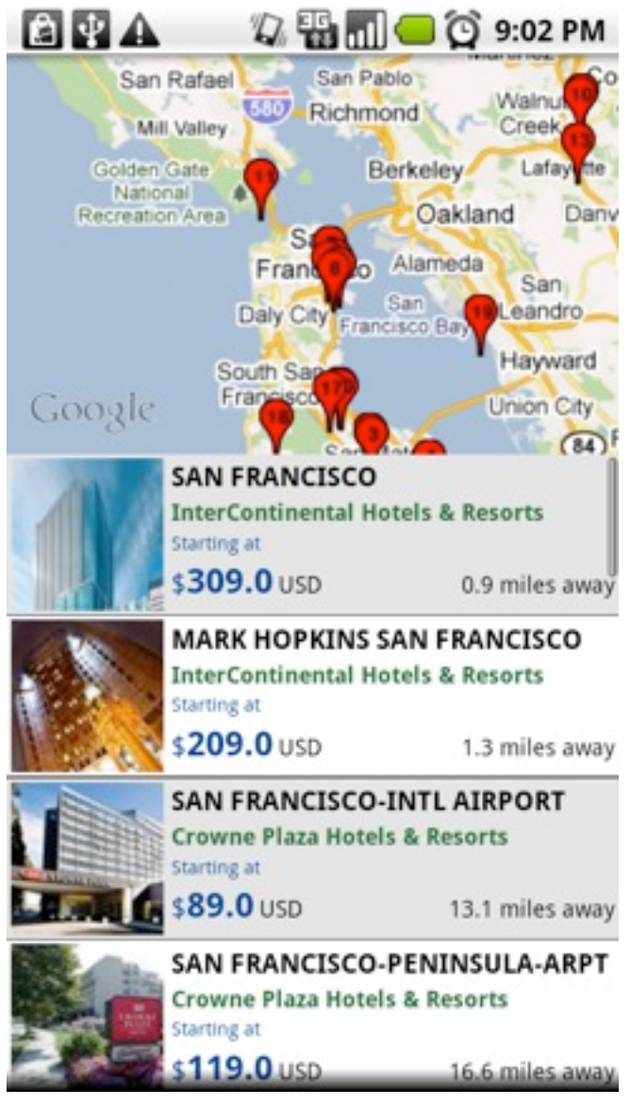
A Case Study: IHG

Google I/O 10

Connecting our Customers with Data

Where are the IHG Hotels?

Android App

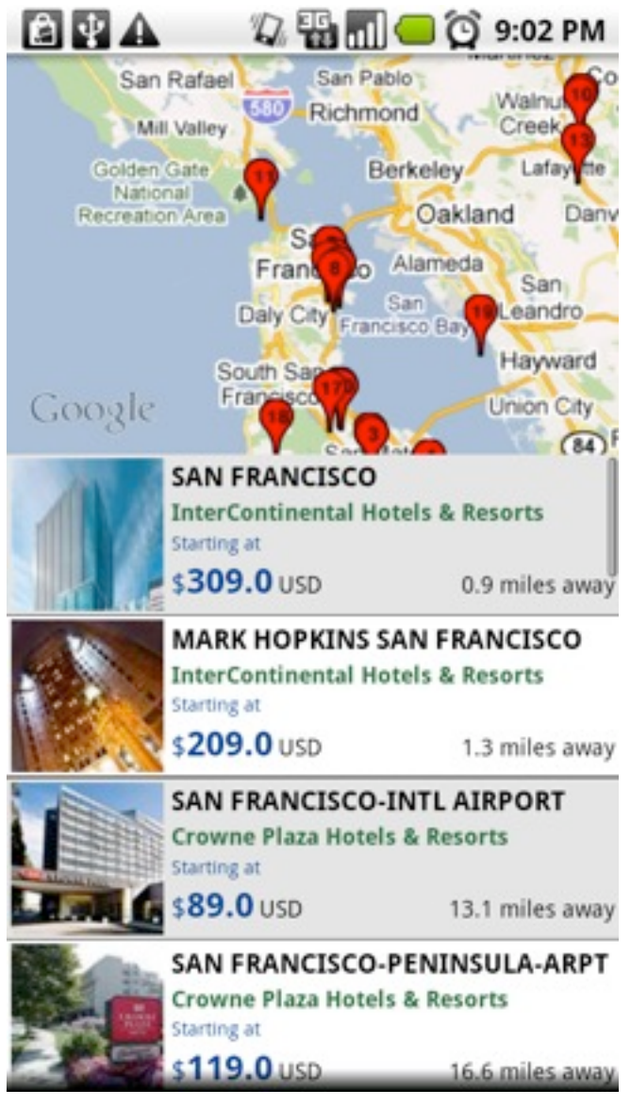


iPhone App

Connecting our Customers with Data

Where are the IHG Hotels?

Android App



iPhone App

A Case Study: IHG

Google I/O 10

Coming Soon!

- JSON Feature Upload/Download!
  - output=jsonc
  - No need to parse KML
  - Perfect match for users of the Javascript client library
- FusionTables in the Javascript V3
  - Structured Data as tables and columns
  - Exposed as a FusionTablesLayer() in V3 (experimental)
  - Provides SQL queries of data
  - MVC means dynamic updating of data

Google™ I/O 10