Geography 485L/585L - Internet Mapping

Karl Benedict - kbene@unm.edu

Spring 2016

Contents

Chapter 1

Week 1 - Module 1 - Introduction and Outline

Overview

- Introductions
- Review of the Syllabus
- Topics to be Covered
- Basics/Definitions

Introductions

- Who am I?
- Who are you?
- What brought you here?

Syllabus Review (link)

Class Topics

- Internet Mapping Clients: Basic HTML, Javascript, CSS; Google Maps API; OpenLayers javascript library
- Geospatial Services Oriented Architectures (SOA)
- Open Standards: Open Geospatial Consortium (OGC - WMS, WFS, WCS, KML); Extensible Markup Language (XML)
- Desktop client use of Open Standards
- Data sharing/publication using Open Standards

Basics

Outline

- What is Internet Mapping?
- Definitions
- Tools

What is Internet Mapping

Extended Desktop Mapping Use of open standards based remote data and map services in desktop applications

Geospatial Data Sharing Establishing open standards based services to share geospatial data and mapping capabilities over the Internet

Web-client Mapping The delivery of mapping and geospatial data tools through web browsers, again based upon open standards

Definitions

Internet The global computer network of computers that typically connect with each other over TCP/IP

World Wide Web The subset of applications that are run over the Internet, typically using the HTTP protocol in combination with data (HTML, XML, XHTML), presentation (CSS), and behavior (JavaScript) components

Mapping The generation of cartographic products that include map images (pictures of geospatial data) and other elements (e.g. legends, tools, scale information, north-arrow)

Definitions

Analysis The development of models (statistical and otherwise) that enable the exploration of geospatial data and testing of hypotheses using those data

Open Standards While the definition varies from one organization to the next, Open Standards are often characterized by the following:

- Developed through a public process by a national or international standards group
- May be implemented royalty-free

Definitions

Interoperability Ability of systems to share data and information with each other

COTS Commercial Off-the-Shelf Software. Applications that are "purchased" from vendors, often with license terms that restrict the use the software to the specific platform for which it is licensed. Often comes with implicit or explicit technical support

Open Source Software licensed under terms that are consistent with the Open Source definition, which includes access to source code, and freedom to modify and redistribute

Definitions

Data Actual values associated with geographic locations. For example - numeric elevation values associated with locations within a Digital Elevation Model.

Metadata Data about a particular data product or service. Metadata provide critical documentation that supports the discovery and use of data products and data and mapping services

Tools

Computer Hardware Requirements

- At least 2 GB RAM
- At least 20 GB of available disk space
- Internet Connection (broadband [>728 Kb/sec] recommended)

Software Requirements

- Supported Operating System
- Geographic Information System (GIS)
- Text Editor
- Secure File Transfer Protocol Client
- Secure Shell (SSH) Client
- Web Browser (at least one of the following)
- A desktop Git/GitHub client for your operating system of choice

Chapter 2

Week 2 - Module 2a - Web-based Mapping Clients. HTML, CSS & Javascript

Overview

- Web Development
- Parts of a web page
- Web Site Components
  - Structure (X/HTML)
  - Presentation (CSS)
  - Behavior (Javascript)
- Simple Web Pages
- More Complete Web Page Example

Web Development

- Requirements
  - Web Server
  - File location that the web server accesses for requested content
  - Files must be readable by all users
- General Process
  - Create basic content in HTML or XHTML (structure)
  - Change appearance of content through the definitions of styles using CSS (presentation)
  - Add dynamic capabilities to content through Javascript (behavior)
  - REPEAT over and over and over and over again

Parts of a Web Page

```
1 <html>
2     <!-- The HTML block is the container for all of your page content -->
3     <head>
```

```
4           <!-- The head is where you include pointers to external resources
5           (i.e. style sheets and javascript files), blocks of Javascript code
6           , styles, etc.  -->
7           <title>The page title also goes in here</title>
8       </head>
9       <body>
10          <!-- The body is where you put all of the content for the page
11          (i.e. the material that will be displayed in the web browser)  -->
12          <h1>Headers</h1>
13          <div>Generic blocks of content</div>
14          <p>Paragraphs</p>
15          <table>Tables</table>
16          <img ...>Images</img>
17          <form ...>Forms</form>
18          <ul>Unordered Lists</ul>
19          <ol>Ordered Lists</ol>
20          <li>List Items</li>
21
22          <!-- Javascript can go here as well -->
23      </body>
24  </html>
```

Link to example/Preview

Web Site Components - Structure

Content is defined in terms of the structural elements available in HTML/XHTML

- Sample HTML/XHTML Tags
  - Paragraphs (i.e. blocks of text) are contained within `<p>...</p>` tags
  - Headings (i.e. section headings, sub-headings) are contained within numerically defined header tags: `<h1>...</h1>, <h2>...</h2>, <h3>...</h3>`, etc.
  - Tabular data are within `<table>...</table>` tags
  - List are specified within `<ol>...</ol>` or `<ul>...</ul>` tags, depending upon whether the list is ordered (numbered) or unordered (e.g. bulleted)
  - User input elements are put within `<form>...</form>` tags
  - Blocks of content (i.e. sections or divisions) are defined within `<div>...</div>` tags
- Structure is translated into the Document Object Model (DOM) for later use by CSS and Javascript

Web Site Components - Presentation

Modifications to default rendering of HTML/XHTML elements are made through styles defined in CSS

- Styles may be
  - defined in an external file that is referenced within the `<head>` block (the preferred method when doing "real" web development)
  - directly defined within the `<head>` block of a web page
  - directly embedded in the elements to which they apply (generally not a "Good Thing")
- When not embedded within an element, a style definition consists of

  - A selector
  - The style definition, enclosed in “curly-brackets”, separated by “semi-colons”
  - For example: `h1 {color:red; font-size:18px;}`

CSS Selectors

Selectors may be based on several criteria

- Element name: `h1`, `p`, `table`, `ul`, etc.
  - Element: `<h1>A top level heading</h1>`
  - Selector: `h1 {color:red; font-size:18px}`
- Element ID: a unique name assigned to HTML/XHTML elements within the structure of the document
  - Element: `<p id="para01">Some text goes here</p>`
  - Selector: `#para01 {color:blue; font-size:12px}`
- Class ID: a name assigned to multiple elements which may be modified through reference to their class
  - Element: `<p class="instructions">Here are some instructions</p>`
  - Another Element: `<p class="instructions">Here are some more instructions</p>`
  - Selector: `.instructions {color:red; font-size:12px; text-decoration:blink}`
- Selectors may be combined in a variety of ways

Web Site Components - Behavior

The most interoperable language for adding dynamic behavior to web sites is *Javascript* - supported by most browsers on most operating systems

- A full-fledged programming language
  - A non-trivial undertaking to become proficient in
  - Experience in other programming languages can contribute to learning Javascript
- Defines actions that may be taken on/by DOM elements
- Allows for modification of existing DOM elements, creation of new DOM elements after the page has finished loading from the server, retrieval of new content after page loads
  - An interactive web page that may behave like a local desktop application

Reference Links

- w3schools.com
  - HTML 4.0 / XHTML 1.0 Tag Reference
  - Cascading Style Sheet (CSS) selectors and elements
  - Javascript reference
- World Wide Web Consortium (W3C)
  - HTML and CSS Background
  - HTML and CSS Tutorial Links Page
  - Validators Page
- Webmonkey.com
  - HTML Cheat Sheet
  - CSS Guide

Simple Web Page

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
        <title>This is a simple web page</title>
    </head>
    <body>
        <h1>They don't get any simpler than this!</h1>
        <p>OK, not much simpler than this.</p>
        <p>Hello World?</p>
    </body>
</html>
```

link to example/Preview

Simple Web Page with CSS

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
        <title>This is a simple web page - with styling</title>
        <style type="text/css">
            h1 {color:blue; font-size:large}
            p.para {color:#777777; font-size:small}
            #annoying {color:red; text-decoration:line-through}
        </style>
    </head>
    <body>
        <h1>They don't get any simpler than this!</h1>
        <p class="para">OK, not much simpler than this.</p>
        <p id="annoying" class="para">Hello World?</p>
    </body>
</html>
```

link to example/Preview

Simple Web Page with Javascript

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
        <title>This is a simple web page with Javascript</title>
        <script type="text/javascript">
            function genericAlert() {
                alert("You just did something ...")
                document.getElementById("clickMe").style.color = "red"
            }
        </script>
    </head>
    <body>
        <h1>They don't get any simpler than this!</h1>
```

```
14          <p>OK, not much simpler than this.</p>
15          <p>Hello World?</p>
16          <p id="clickMe" onclick="genericAlert();">What happens when you click me?</p>
17      </body>
18  </html>
```

link to example/Preview

More Complete Web Page Example

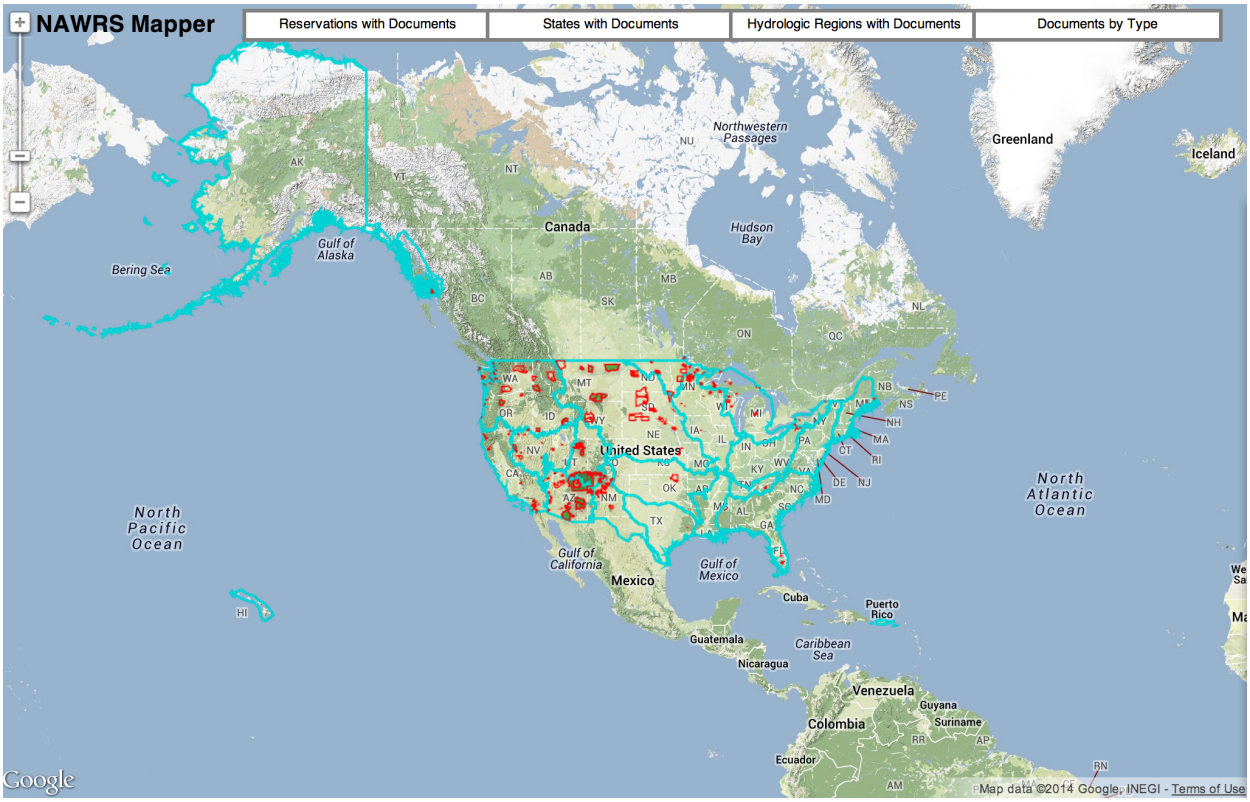


Figure 2.1: NAWRS Mapper. *HTML*: 39 Lines; *CSS*: 136 Lines; *core.js*: 515 Lines + Google Maps API and JQuery Framework

Chapter 3

Week 3- Module 2a - Web-based Mapping Clients. Google Maps API

Outline

- What is an API
- The Google Maps API
  - Version
  - Reference Information
  - Key Components
  - Examples

What is an API

- API Stands for Application Programming Interface

  An Application Programming Interface (API) is a particular set of rules and specifications that a software program can follow to access and make use of the services and resources provided by another particular software program that implements that API. It serves as an interface between different software programs and facilitates their interaction, similar to the way the user interface facilitates interaction between humans and computers. – From Wikipedia: http://en.wikipedia.org/wiki/Api

- The Google Maps API provides an interface for interacting with Google's mapping services from external web applications

Google Maps API Version

- The version of the Google Maps API used in this class is v3 of the Javascript API
  - Freely usable for free applications
  - Subject to Google's Terms of Service
  - Google API key is optional for our work
- Key capabilities in v3
  - Interactive maps based on Google's mapping engine (contrast w. static maps API)
  - Optimized for desktop and mobile platforms and applications

Reference Information

Google Maps API Family http://code.google.com/apis/maps/
Javascript API Home Page http://code.google.com/apis/maps/documentation/javascript/
Javascript API v3 Tutorial Page http://code.google.com/apis/maps/documentation/javascript/tutorial.html

Key Components

- Map object options

Types (required) ROADMAP
SATELLITE
HYBRID
TERRAIN
Latitude and Longitude (required) specification of where the map should initially be centered
Zoom Level (required) 0=global, higher values increasingly local. Limited by map type

Controls

- Available Controls (enabled through map options) default controls
  - Zoom Control
  - Pan Control
  - Scale Control
  - MapType Control
  - Street View Control
  - Rotate (for maps that contain 45-degree imagery)
- Different control styles may be defined
- Controls may be positioned positioning options
- Custom controls may be defined and attached to fixed location in the map

Overlays

Overlay Types documentation

Marker points depicted by specified or defined icons at locations within the map
Polyline linear features defined by multiple points with a defined style for the line
Polygon closed features defined by multiple points. Supports multi-polygons, and donuts. Line and fill styles may be specified.
(Ground) Overlay Maps Image-based map layers that replace or overlay Google layers - registered to the map coordinates

Overlays (continued)

Info Windows floating content windows for displaying content defined as HTML, a DOM element, or text string
Layers Grouped display content assigned to a specific layer type: Data (including GeoJSON), KmlLayer (& GeoRSS), Heatmap, FusionTablesLayer, TrafficLayer, TransitLayer BicyclingLayer
Custom Overlays definition of programmatically controlled layers

Services

- Geocoding Service
  - Forward and reverse geocoding:
    - address to LatLon
    - LatLon to Nearest Address
  - May be biased to current viewport, region
- Directions
  - Based upon an origin, destination, and a variety of additional options
  - Available directions and rendered route
- Distance Matrix
  - Travel distance and duration given a specific mode of travel

Services

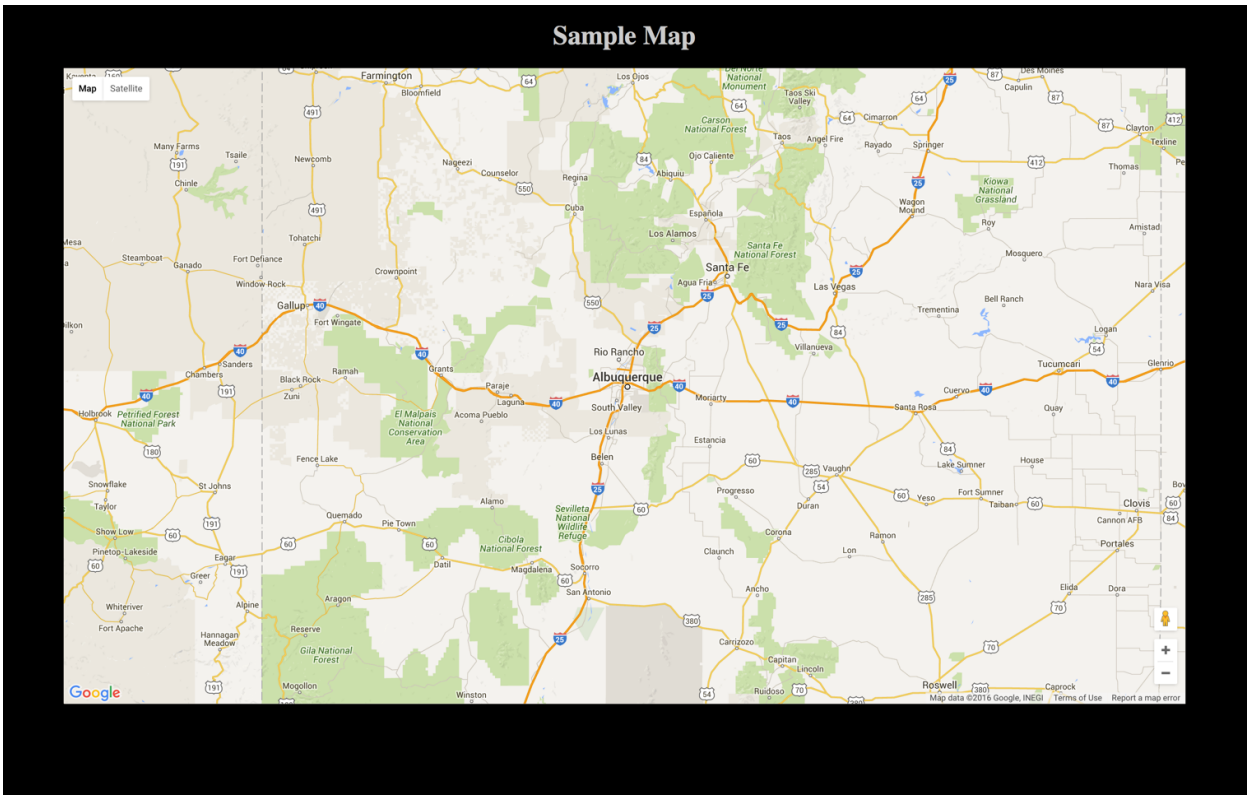
- Elevation
  - Delivery of elevation data for locations or paths
- Streetview
  - Integration of Google Streetview within a DOM element
- Maximum Zoom
  - Provides information about the maximum available zoom level

Events

- Events provide the ability to attach custom behaviors to events in the interface. For example:
  - Changing items in the interface as the user zooms in on a map
  - Displaying additional information outside the map when the user clicks a location in the map
  - Synchronizing the behavior of multiple maps as the user interacts with one map
- Requires higher-level Javascript than we will cover in this course

Examples

Simple - Roadmap



Simple - Roadmap Code

gmaps01.html

```
<!DOCTYPE html>
<html>
    <head>
        <link rel="stylesheet" type="text/css" href="css/mapPage.css">
    </head>

    <body>
        <h1>Sample Map</h1>
        <div id="map_canvas"></div>

    <!-- Let's put our JavaScript down here ----------------------------------------->
        <!-- Load the external JavaScript file with the map definition code -->
        <script src="js/mapPage_01.js"></script>

        <!-- Load the API in asynchronous mode and execute the initialize
            function when done -->
        <!-- The optional 'key-<API Key>' parameter is not used here but should be
            for a production map -->
        <script async defer
```

```
            src="https://maps.googleapis.com/maps/api/js?callback=initialize">
        </script>
    </body>
</html>
```

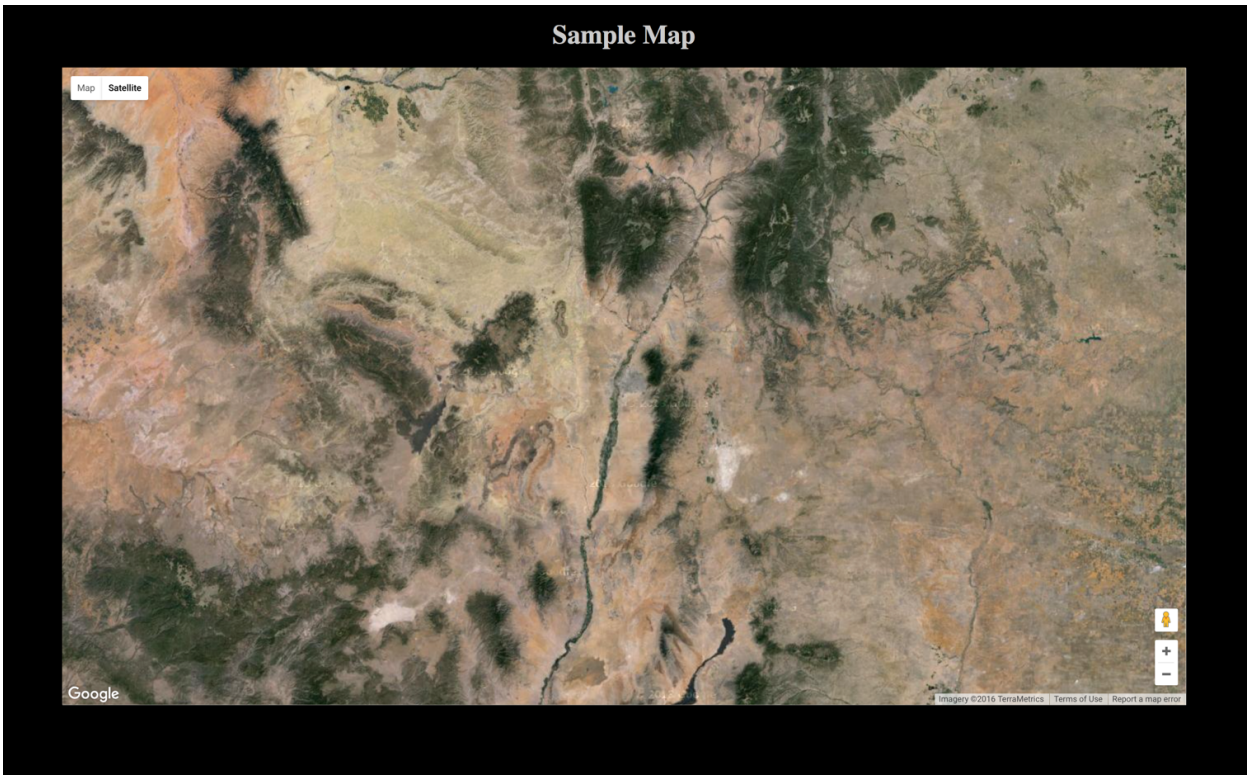
mapPage.css

```
/* You must set the height of either the 'html' or 'body' elements for some
    browsers to properly render the map with a hight taller than 0px */
html {
    height: 100% }

body {
    height: 100%;
    margin: 0px;
    padding: 0px;
    background-color: black;
    color: #CCCCCC;
    text-align: center}

#map_canvas {
    width:90%;
    height:80%;
    margin-left:auto;
    margin-right: auto }

.infoBox {
    color:black }
```

mapPage_01.js

```
function initialize() {
    var classroom = new google.maps.LatLng(35.084280,-106.624073)
    var mapOptions = {
        zoom: 8,
        center: classroom,
        mapTypeId: google.maps.MapTypeId.ROADMAP
    };
    var map = new google.maps.Map(
        document.getElementById("map_canvas"),
        mapOptions);
}

```

Simple - Satellite



Simple - Satellite Code

gmaps02.html

```
<!DOCTYPE html>
<html>
    <head>
        <link rel="stylesheet" type="text/css" href="css/mapPage.css">
    </head>

    <body>
        <h1>Sample Map</h1>
        <div id="map_canvas"></div>

    <!-- Let's put our JavaScript down here ----------------------------------------->
        <!-- Load the external JavaScript file with the map definition code -->
        <script src="js/mapPage_02.js"></script>

        <!-- Load the API in asynchronous mode and execute the initialize
            function when done -->
        <!-- The optional 'key-<API Key>' parameter is not used here but should be
            for a production map -->
        <script async defer
            src="https://maps.googleapis.com/maps/api/js?callback=initialize">
        </script>
```

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

mapPage_02.js

```
function initialize() {
    var classroom = new google.maps.LatLng(35.084280,-106.624073)
    var mapOptions = {
        zoom: 8,
        center: classroom,
        mapTypeId: google.maps.MapTypeId.SATELLITE
    };
    var map = new google.maps.Map(
        document.getElementById("map_canvas"),
        mapOptions);
}

```

Simple - Hybrid

Simple - Hybrid Code

gmaps03.html

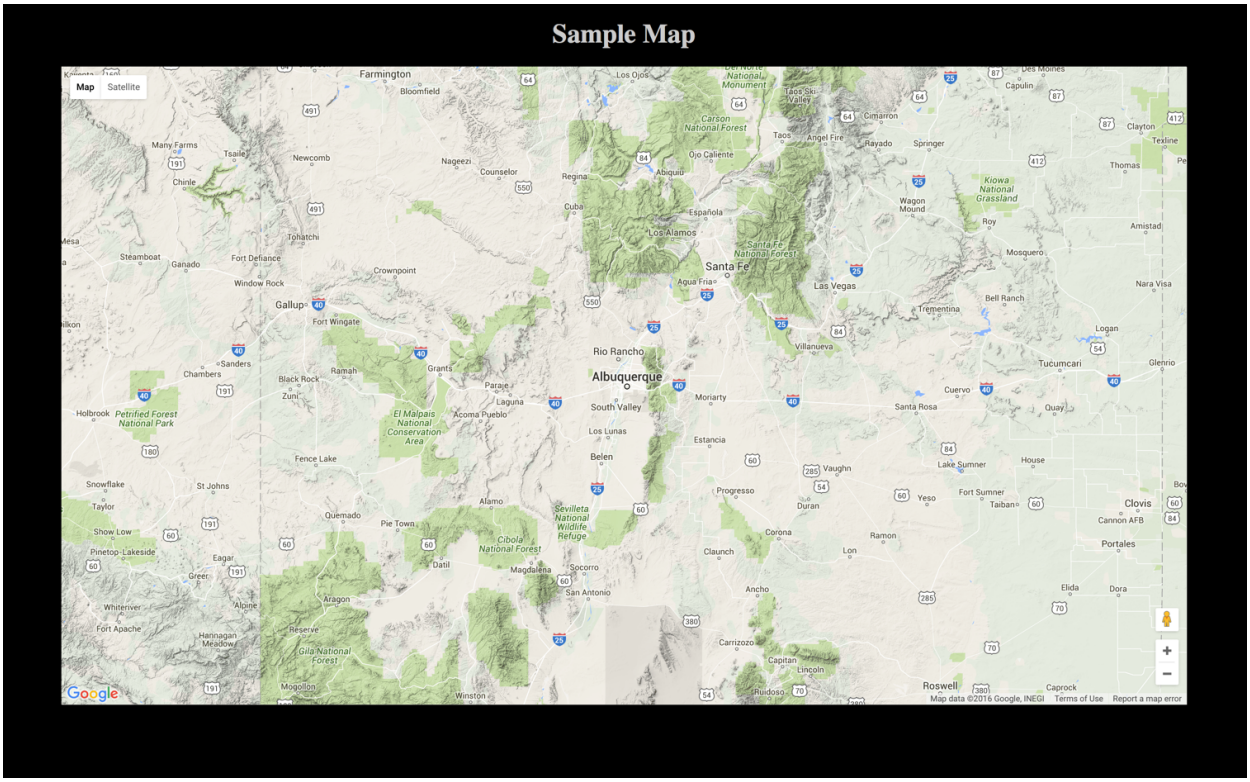
```
<!DOCTYPE html>
<html>
```

```
    <head>
        <link rel="stylesheet" type="text/css" href="css/mapPage.css">
    </head>

    <body>
        <h1>Sample Map</h1>
        <div id="map_canvas"></div>

    <!-- Let's put our JavaScript down here ----------------------------------------->
        <!-- Load the external JavaScript file with the map definition code -->
        <script src="js/mapPage_03.js"></script>

        <!-- Load the API in asynchronous mode and execute the initialize
            function when done -->
        <!-- The optional 'key-<API Key>' parameter is not used here but should be
            for a production map -->
        <script async defer
            src="https://maps.googleapis.com/maps/api/js?callback=initialize">
        </script>
    </body>
</html>
```

mapPage_03.js

```
function initialize() {
    var classroom = new google.maps.LatLng(35.084280,-106.624073)
    var mapOptions = {
        zoom: 8,
        center: classroom,
        mapTypeId: google.maps.MapTypeId.HYBRID
    };
    var map = new google.maps.Map(
        document.getElementById("map_canvas"),
        mapOptions);
}

```

Simple - Terrain



Simple - Terrain Code

gmaps04.html

```
<!DOCTYPE html>
<html>
    <head>
        <link rel="stylesheet" type="text/css" href="css/mapPage.css">
    </head>

    <body>
        <h1>Sample Map</h1>
        <div id="map_canvas"></div>

    <!-- Let's put our JavaScript down here ----------------------------------------->
        <!-- Load the external JavaScript file with the map definition code -->
        <script src="js/mapPage_04.js"></script>

        <!-- Load the API in asynchronous mode and execute the initialize
            function when done -->
        <!-- The optional 'key-<API Key>' parameter is not used here but should be
            for a production map -->
        <script async defer
            src="https://maps.googleapis.com/maps/api/js?callback=initialize">
        </script>
```

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

mapPage_04.js

```
function initialize() {
    var classroom = new google.maps.LatLng(35.084280,-106.624073)
    var mapOptions = {
        zoom: 8,
        center: classroom,
        mapTypeId: google.maps.MapTypeId.TERRAIN
    };
    var map = new google.maps.Map(
        document.getElementById("map_canvas"),
        mapOptions);
}

```

Simple - Hybrid - Zoomed

Simple - Hybrid - Zoomed Code

gmaps05.html

```
<!DOCTYPE html>
<html>
```

```
    <head>
        <link rel="stylesheet" type="text/css" href="css/mapPage.css">
    </head>

    <body>
        <h1>Sample Map</h1>
        <div id="map_canvas"></div>

    <!-- Let's put our JavaScript down here ----------------------------------------->
        <!-- Load the external JavaScript file with the map definition code -->
        <script src="js/mapPage_05.js"></script>

        <!-- Load the API in asynchronous mode and execute the initialize
            function when done -->
        <!-- The optional 'key-<API Key>' parameter is not used here but should be
            for a production map -->
        <script async defer
            src="https://maps.googleapis.com/maps/api/js?callback=initialize">
        </script>
    </body>
</html>
```

mapPage_05.js

```
function initialize() {
    var classroom = new google.maps.LatLng(35.084280,-106.624073)
    var mapOptions = {
        zoom: 18,
        center: classroom,
        mapTypeId: google.maps.MapTypeId.TERRAIN
    };
    var map = new google.maps.Map(
        document.getElementById("map_canvas"),
        mapOptions);
}

```

Simple - Zoomed - Modified Controls

Simple - Zoomed - Modified Controls Code

gmaps06.html

```
<!DOCTYPE html>
<html>
    <head>
        <link rel="stylesheet" type="text/css" href="css/mapPage.css">
    </head>

    <body>
        <h1>Sample Map</h1>
        <div id="map_canvas"></div>

    <!-- Let's put our JavaScript down here ----------------------------------------->
        <!-- Load the external JavaScript file with the map definition code -->
        <script src="js/mapPage_06.js"></script>

        <!-- Load the API in asynchronous mode and execute the initialize
            function when done -->
        <!-- The optional 'key-<API Key>' parameter is not used here but should be
            for a production map -->
        <script async defer
            src="https://maps.googleapis.com/maps/api/js?callback=initialize">
        </script>
```

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

mapPage_06.js

```
function initialize() {
var classroom = new google.maps.LatLng(35.084280,-106.624073)
var myOptions = {
    zoom: 18,
    center: classroom,
    mapTypeId: google.maps.MapTypeId.HYBRID,
    zoomControl: true,
    zoomControlOptions: {style: google.maps.ZoomControlStyle.SMALL},
    mapTypeControl: true,
    mapTypeControlOptions: {
    style: google.maps.MapTypeControlStyle.DROPDOWN_MENU},
    streetViewControl: false
};
var map = new google.maps.Map(
    document.getElementById("map_canvas"),
    myOptions);
}

```

Markers

Markers Code

gmaps07.html

```
<!DOCTYPE html>
<html>
    <head>
        <link rel="stylesheet" type="text/css" href="css/mapPage.css">
    </head>

    <body>
        <h1>Sample Map</h1>
        <div id="map_canvas"></div>

    <!-- Let's put our JavaScript down here ----------------------------------------->
        <!-- Load the external JavaScript file with the map definition code -->
        <script src="js/mapPage_07.js"></script>

        <!-- Load the API in asynchronous mode and execute the initialize
            function when done -->
        <!-- The optional 'key-<API Key>' parameter is not used here but should be
            for a production map -->
        <script async defer
            src="https://maps.googleapis.com/maps/api/js?callback=initialize">
        </script>
    </body>
</html>
```

mapPage_07.js

```
function initialize() {
    var classroom = new google.maps.LatLng(35.084280,-106.624073)
    var office = new google.maps.LatLng(35.084506,-106.624899)
    var myOptions = {
        zoom: 18,
        center: classroom,
        mapTypeId: google.maps.MapTypeId.HYBRID
        };
    var map = new google.maps.Map(
        document.getElementById("map_canvas"),
        myOptions);

    var classroomMarker = new google.maps.Marker({
        position: classroom,
        title:"Geography 485L/585L Classroom, Bandelier East, Room 106"
        });
    classroomMarker.setMap(map);

    var officeMarker = new google.maps.Marker({
        position: office,
        title:"Office, Bandelier West, Room 107"
        });
    officeMarker.setMap(map);
```

```
}

```

Polyline

Polyline Code

gmaps08.html

```
<!DOCTYPE html>
<html>
    <head>
        <link rel="stylesheet" type="text/css" href="css/mapPage.css">
    </head>

    <body>
        <h1>Sample Map</h1>
        <div id="map_canvas"></div>

    <!-- Let's put our JavaScript down here ----------------------------------------->
        <!-- Load the external JavaScript file with the map definition code -->
        <script src="js/mapPage_08.js"></script>

        <!-- Load the API in asynchronous mode and execute the initialize
            function when done -->
        <!-- The optional 'key-<API Key>' parameter is not used here but should be
            for a production map -->
```

```
        <script async defer
            src="https://maps.googleapis.com/maps/api/js?callback=initialize">
        </script>
    </body>
</html>
```

mapPage_08.js

```
    var classroom = new google.maps.LatLng(35.084280,-106.624073)
    var office = new google.maps.LatLng(35.084506,-106.624899)
    var myOptions = {
        zoom: 18,
        center: classroom,
        mapTypeId: google.maps.MapTypeId.HYBRID
        };
    var map = new google.maps.Map(
        document.getElementById("map_canvas"),
        myOptions);

    var classroomMarker = new google.maps.Marker({
        position: classroom,
        title:"Geography 485L/585L Classroom, Bandelier East, Room 106"
        });
    classroomMarker.setMap(map);

    var officeMarker = new google.maps.Marker({
        position: office,
        title:"Office, Bandelier West, Room 107"
        });
    officeMarker.setMap(map);

    var officeVisitCoordinates = [
        office,
        new google.maps.LatLng(35.084445,-106.624327),
        new google.maps.LatLng(35.084309,-106.624308),
        classroom
        ];
    var officePath = new google.maps.Polyline({
        path: officeVisitCoordinates,
        strokeColor: "#FF0000",
        strokeOpacity: 1.0,
        strokeWeight: 2
    });
    officePath.setMap(map)
}

```

Polygon

Polygon Code

gmaps09.html

```
<!DOCTYPE html>
<html>
    <head>
        <link rel="stylesheet" type="text/css" href="css/mapPage.css">
    </head>

    <body>
        <h1>Sample Map</h1>
        <div id="map_canvas"></div>

    <!-- Let's put our JavaScript down here ----------------------------------------->
        <!-- Load the external JavaScript file with the map definition code -->
        <script src="js/mapPage_09.js"></script>

        <!-- Load the API in asynchronous mode and execute the initialize
            function when done -->
        <!-- The optional 'key-<API Key>' parameter is not used here but should be
            for a production map -->
        <script async defer
            src="https://maps.googleapis.com/maps/api/js?callback=initialize">
        </script>
```

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

mapPage_09.js

```
function initialize() {
    var classroom = new google.maps.LatLng(35.084280,-106.624073)
    var office = new google.maps.LatLng(35.084506,-106.624899)
    var myOptions = {
        zoom: 18,
        center: classroom,
        mapTypeId: google.maps.MapTypeId.HYBRID
        };
    var map = new google.maps.Map(
        document.getElementById("map_canvas"),
        myOptions);
    var classroomMarker = new google.maps.Marker({
        position: classroom,
        title:"Geography 485L/585L Classroom, Bandelier East, Room 106"
        });
    classroomMarker.setMap(map);
    var officeMarker = new google.maps.Marker({
        position: office,
        title:"Office, Bandelier West, Room 107"
        });
    officeMarker.setMap(map);
    var buildingCoordinates = [
        new google.maps.LatLng(35.084498,-106.624921),
        new google.maps.LatLng(35.084558,-106.624911),
        new google.maps.LatLng(35.084566,-106.624970),
        new google.maps.LatLng(35.084609,-106.624966),
        new google.maps.LatLng(35.084544,-106.624383),
        new google.maps.LatLng(35.084438,-106.624317),
        new google.maps.LatLng(35.084384,-106.623922),
        new google.maps.LatLng(35.084164,-106.623970),
        new google.maps.LatLng(35.084214,-106.624324),
        new google.maps.LatLng(35.084214,-106.624324),
        new google.maps.LatLng(35.084391,-106.624284)
        ];
    var bldgPoly = new google.maps.Polygon({
        paths: buildingCoordinates,
        strokeColor: "#FF0000",
        strokeOpacity: 0.8,
        strokeWeight: 2,
        fillColor: "#FF0000",
        fillOpacity: 0.35
    });
    bldgPoly.setMap(map)
}

```

Adding an Info Window

Adding an Info Window Code

gmaps10.html

```
<!DOCTYPE html>
<html>
    <head>
        <link rel="stylesheet" type="text/css" href="css/mapPage.css">
    </head>

    <body>
        <h1>Sample Map</h1>
        <div id="map_canvas"></div>

    <!-- Let's put our JavaScript down here ----------------------------------------->
        <!-- Load the external JavaScript file with the map definition code -->
        <script src="js/mapPage_10.js"></script>

        <!-- Load the API in asynchronous mode and execute the initialize
            function when done -->
        <!-- The optional 'key-<API Key>' parameter is not used here but should be
            for a production map -->
        <script async defer
            src="https://maps.googleapis.com/maps/api/js?callback=initialize">
        </script>
```

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

mapPage_10.js

```
function initialize() {
    var classroom = new google.maps.LatLng(35.084280,-106.624073)
    var office = new google.maps.LatLng(35.084506,-106.624899)
    var myOptions = {
        zoom: 18,
        center: classroom,
        mapTypeId: google.maps.MapTypeId.HYBRID
        };
    var map = new google.maps.Map(
        document.getElementById("map_canvas"),
        myOptions);
    var classroomMarker = new google.maps.Marker({
        position: classroom,
        title:"Geography 485L/585L Classroom, Bandelier East, Room 106"
        });
    classroomMarker.setMap(map);
    var officeMarker = new google.maps.Marker({
        position: office,
        title:"Office, Bandelier West, Room 107"
        });
    officeMarker.setMap(map);
    var buildingCoordinates = [
        new google.maps.LatLng(35.084498,-106.624921),
        new google.maps.LatLng(35.084558,-106.624911),
        new google.maps.LatLng(35.084566,-106.624970),
        new google.maps.LatLng(35.084609,-106.624966),
        new google.maps.LatLng(35.084544,-106.624383),
        new google.maps.LatLng(35.084438,-106.624317),
        new google.maps.LatLng(35.084384,-106.623922),
        new google.maps.LatLng(35.084164,-106.623970),
        new google.maps.LatLng(35.084214,-106.624324),
        new google.maps.LatLng(35.084214,-106.624324),
        new google.maps.LatLng(35.084391,-106.624284)
        ];
    var bldgPoly = new google.maps.Polygon({
        paths: buildingCoordinates,
        strokeColor: "#FF0000",
        strokeOpacity: 0.8,
        strokeWeight: 2,
        fillColor: "#FF0000",
        fillOpacity: 0.35
        });
    bldgPoly.setMap(map);
    var classInfoContent = '<div class="infoBox">' +
        '<p>This is the location for the Geography 485L/585L class</p>' +
        '</div>'
    var classInfoWindow = new google.maps.InfoWindow({
        content: classInfoContent
        });
```

```
50      google.maps.event.addListener(classroomMarker, 'click', function() {
51          classInfoWindow.open(map,classroomMarker);
52          });
53  }
54
```
