Lab 5 Geospatial Web services

Total points: 50

Introduction

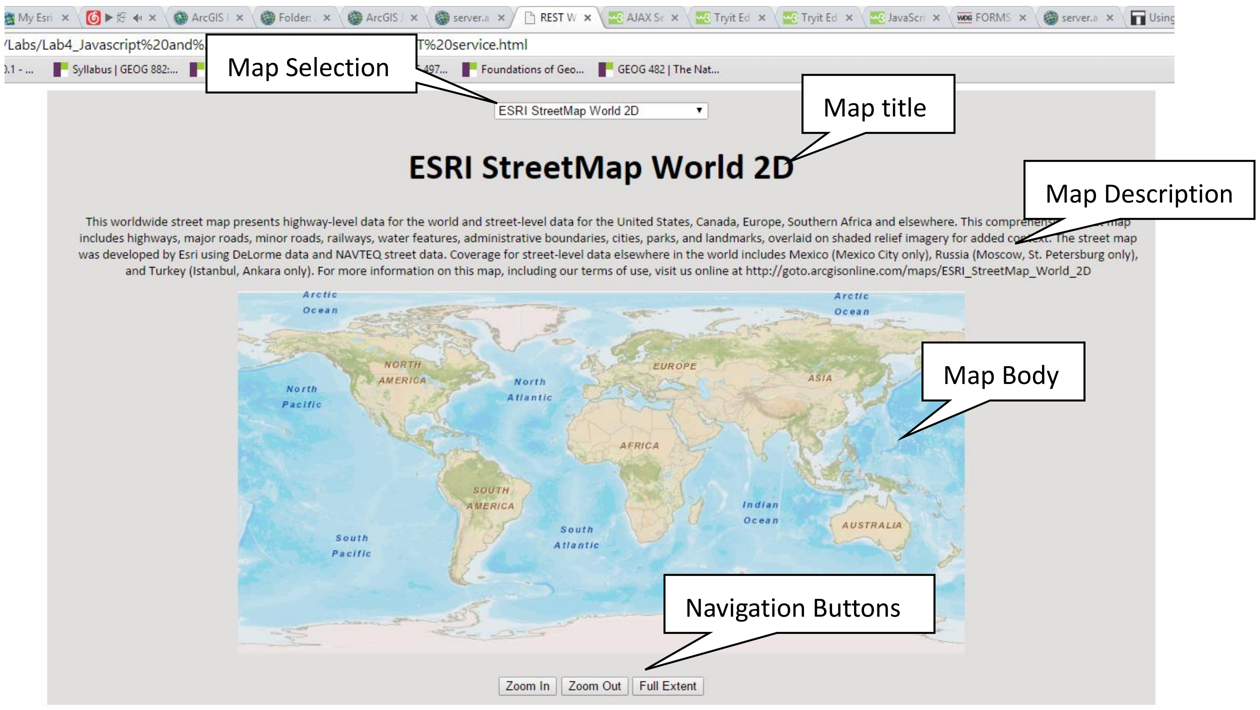
This lab asks you to develop a web page showing three maps using the ArcGIS Online web services.

- **ESRI_StreetMap_World_2D** (http://server.arcgisonline.com/arcgis/rest/services/ESRI_StreetMap_World_2D/MapServer)
- **USA_Topo_Maps** (http://server.arcgisonline.com/arcgis/rest/services/USA_Topo_Maps/MapServer)
- **Demographics/USA_1990-2000_Population_Change** (http://server.arcgisonline.com/arcgis/rest/services/Demographics/USA_1990-2000_Population_Change/MapServer)

The purpose of this lab is to help you gain a good understanding of how to use JavaScript programming and the ArcGIS REST API by implementing very basic Web GIS navigation functions. You will create this Web page from the scratch. In the future, we are going to use ArcGIS API for JavaScript (library), which makes the JavaScript programming much easier than this.

The general requirements are as below:

1. **The layout of the web page should look similar to the following graph**



ESRI StreetMap World 2D

This worldwide street map presents highway-level data for the world and street-level data for the United States, Canada, Europe, Southern Africa and elsewhere. This comprehens [illegible] map includes highways, major roads, minor roads, railways, water features, administrative boundaries, cities, parks, and landmarks, overlaid on shaded relief imagery for added context. The street map was developed by Esri using DeLorme data and NAVTEQ street data. Coverage for street-level data elsewhere in the world includes Mexico (Mexico City only), Russia (Moscow, St. Petersburg only), and Turkey (Istanbul, Ankara only). For more information on this map, including our terms of use, visit us online at http://goto.arcgisonline.com/maps/ESRI_StreetMap_World_2D

The required contents include:

a) A drop-down list (*<select >*and *<option>* HTML tags) showing the three choices of the web maps

b) The title of the selected Web map

c) The description of the selected Web map (*DYNAMICALLY extracted from the metadata of the web service instead of pre-writing in the HTML codes*)

d) Map body (*size: width = 800px, height = 400px*)

e) Three navigation buttons (*<button>* HTML tag):

- Full Extent*: Show the full extent of the map (The full extent bounding box info should be extracted from the metadata of the web service)*
- Zoom In*: The map will be zoomed in by 2 times each time when it is clicked. The map center should remain unchanged.*
- Zoom Out*: The map will be zoomed out by 2 times each time when it is clicked. The map center should remain unchanged.*

2. **Use the objects and functions in the provided JavaScript file (*HTTPRequest.js*) to send a request to a server with a URL and then process the response.**

   a) You have used the same JavaScript file to do the similar work in lab 4 Task 2. You need to link this file into your HTML web page before you use any functions in it.

   b) Create another JavaScript file where you will use ***sendHttpRequest (URL)*** function when you need to send a request to a server with a URL. You have to create and complete your logic in a function called ***handleResponseData (xmlResponse)***, which will be called automatically by the browser to process the response (i.e., *xmlResponse* parameter) when it is successfully sent to your web browser.

   c) Link both the *HTTPRequest.js* file and your own JavaScript file to your HTML Web page.

3. **The JavaScript codes, CSS codes, and HTML codes should be placed on separate files and linked with each other.**

4. **Some tips for implementation**

   a) When you select a map from the drop-down list or click any map navigation button, you are going to use JavaScript to construct the corresponding URL (including the required operation and parameters) for that Web service and use ***sendHttpRequest (URL)*** function to send the request. The response from the server will be the metadata of the map service in JSON format (long text). You need to parse it in JavaScript codes to extract the required information, such as the map description and initial extent of the map.

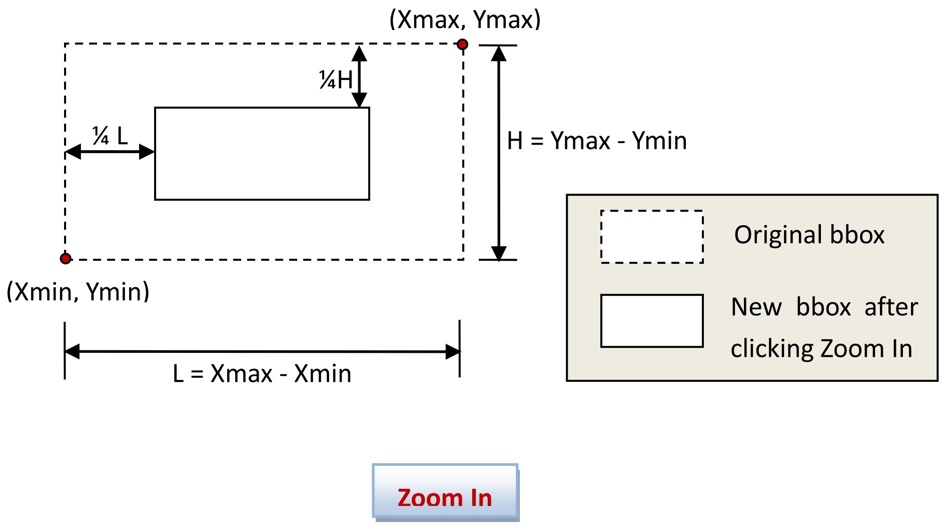
b) The map titles can be hard coded (i.e., pre-created) in the *select* HTML element. The attribute optMapList.options[optMapList.selectedIndex].text *(optMapList is assumed to be the variable representing the select HTML element)* can return the text of the currently selected option in the drop-down list (i.e., current web map title).

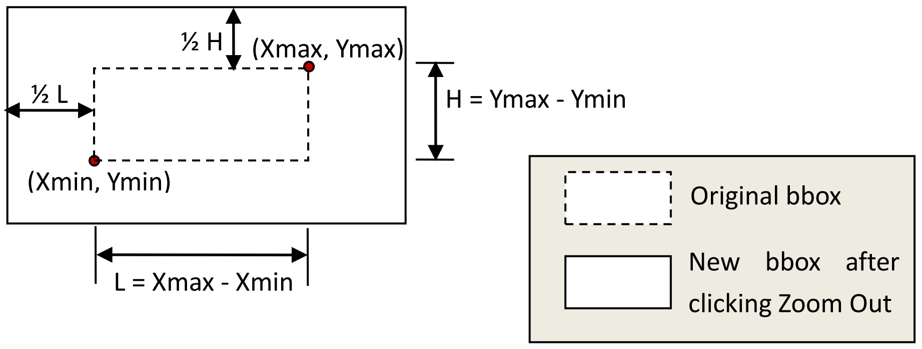
c) The map description about the selected Web map description and its initial/full extent (*XMin*, *XMax*, *YMin*, and *YMax*) should be obtained from the metadata of the Web service (i.e., Web service description).

d) The map body actually is an image returned from the server. Every time the map body updates (e.g., zoom in/out), a new image will be requested and the ***src*** property of the image element on the web page should be updated using JavaScript. If you choose the output format as ***image*** in your URL for requesting the web service, you can just assign the ***src*** property with the URL with parameters. That will send the HTTP request without using ***sendHttpRequest (URL)*** function. Some special characters may not be allowed in the URL, such as space. Use predefined ***encodeURI (original URL)*** JavaScript function that will convert all special characters in a URL to expressions that can be accepted.

e) The key of the three map navigation functions is to calculate the correct map extent to request (i.e. ***bbox***). The center of the map should remain unchanged. You need to use a set of coordinates (xmin, ymin, xmax, and ymax) to hold the current map extent so that you can use them to calculate the new map extent when the user clicks any navigation button.

The following figure shows the relationship between the original bbox and the new bbox after clicking *Zoom In* or *Zoom Out* with a ratio of 2 in this lab.





5. **Grading rubric:**

    a) All required HTML contents are shown on the web page. **(5)**

    b) The layout and style of the HTML elements are displayed as required **(5)**

    c) Map title, description ,and map body can change correspondingly when different map is selected from the drop-down list **(13)**

    d) Three navigation functions work properly **(24)**

    e) Separate CSS, JavaScript, and HTML codes **(3)**

Deliverables:

Compress your HTML, CSS, and JavaScript files in a single zip file and upload it to the Canvas.