Geography 485L/585L Weekly Breakdown

Karl Benedict

Spring 2016

Contents

Chapter 1

Goals and Objectives

Internet mapping technologies are an important component of geospatial data capture, sharing, visualization, and delivery. This course provides a survey of current and emerging internet and geospatial interoperability standards, technologies, and capabilities. The emphasis of the work in this class will be hands-on experience in four critical aspects of Internet-enabled mapping:

- The basic concepts behind web development and web mapping technologies that enable the delivery of maps and mapped data through web browsers
- The Open Standards that facilitate the exchange of map images and geospatial data over the internet
- The use of published standards-based services in desktop mapping applications that implement those standards
- The deployment of standards-based geospatial map and data services that other systems and users may make use of

The specific class objectives that relate to these activities and departmental curriculum objectives for undergraduate and graduate students in the Geography Department include the following:

- Students will understand the concepts geospatial data and service interoperability
- Students will be able to define the specific requirements of a particular analysis or project and identify the interoperability standards that are capable of meeting those requirements
- Students will be knowledgeable in the core technologies that they may use to produce their own internet-enabled mapping capabilities
- Students will understand the strengths and limitations of current internet mapping technologies for generating cartographically effective map products.

The weekly goals, objectives and assignments are outlined below

Chapter 2

Week 1 - Introductions, Course Outline & Web Concepts

This week we will review the content and structure for the course and spend some time getting to know each other. Following this we will spend some time setting up some of the tools that you will be using for the course in developing your portfolio of materials.

Class Prep

- Wikipedia article - History of the World Wide web
- Lynda.com tutorials
  - *Web Design Fundamentals*
    - Introduction
    - 1. Exploring Web Design
  - *Version Control for Everyone*
    - Introduction
    - 1. Introducing Version Control
    - 2. Version Control Basics
    - 3. Setting Up Your First Project

Reference Materials

Class Syllabus

Weekly Milestone - Creating Your GitHub Repository and First Web Page

Developing content to go onto the web has evolved from a solitary effort to one where teams work together in developing components of larger web sites. These teams need to have a variety of tools to enable their work. Some of the most important tools enable code sharing with the team, and in projects based on the Open

Source software model the rest of the world. The GitHub web platform uses the Git distributed version control system to enable sharing of code and hosting static web pages based on that shared code.

You will be using a private GitHub repository to build your class portfolio during the course. If you would like to make your portfolio available publicly you can also use GitHub as the platform for providing that public access. Regardless of your decision about providing public access to your portfolio, you will learn how version control operates, and how to provide comments and keep notes on your work and comment on the work of others (this will be part of our peer review process).

While the work we do this and next week will be directly through the editor integrated into the GitHub system, you will eventually need to install a desktop application (such as the SourceTree application recommended for the class) that allows you to develop your web pages on your local computer and then update the files on the GitHub system when you want to share a new version. Also, you can't add things like images to your web pages until you are adding them to a local repository on your computer and then sending them GitHub.

For this milestone we will walk through the process of creating your repository in GitHub, creating your first web page, previewing that page on your local computer, changing the page, and updating the page on LoboGit. For this milestone we will do this as a manual process which we will streamline in the coming weeks.

Step 1 - Create Your GitHub Account and Portfolio Repository

For your work in this class you will build your portfolio within an organization (https://github.com/UNM-GEOG-485-585) within GitHub that has been created for the class. The first step in the process of creating your portfolio is to create a new *repository* in GitHub within which you will put your portfolio materials for sharing within the class. Please follow the following steps to create your repository:

1. Go to the GitHub homepage and follow the onscreen instructions for creating a new account. If you already have an account you can skip this step.
2. Come to the front of the class and tell me your GitHub username so that I can add you to the organization and create your repository for you within the organization.

Step 2 - Create Your First Web Page

To create your first web page within your portfolio repository you need to first enter your repository, add a new file, modify its contents, and commit your modifications back to the repository to save your changes.

1. Go to the class organization page ((https://github.com/UNM-GEOG-485-585 - logging in if necessary) and click on your repository name in the list.
2. On the page that comes up listing the files in your repository, click the "New File" button above the list of files.

3. Enter the name of the file that you are creating as "hello-world.html"
4. Enter the following text into the text entry area under the filename field.

```html
<html>
    <head>
    </head>
    <body>
        Hello World !!!
    </body>
</html>
```

5. Add a brief comment (such as "Created hello-world.html from provided text") in the first field under the "Commit new file" title. You can optionally add a more detailed description in the next field if you like.

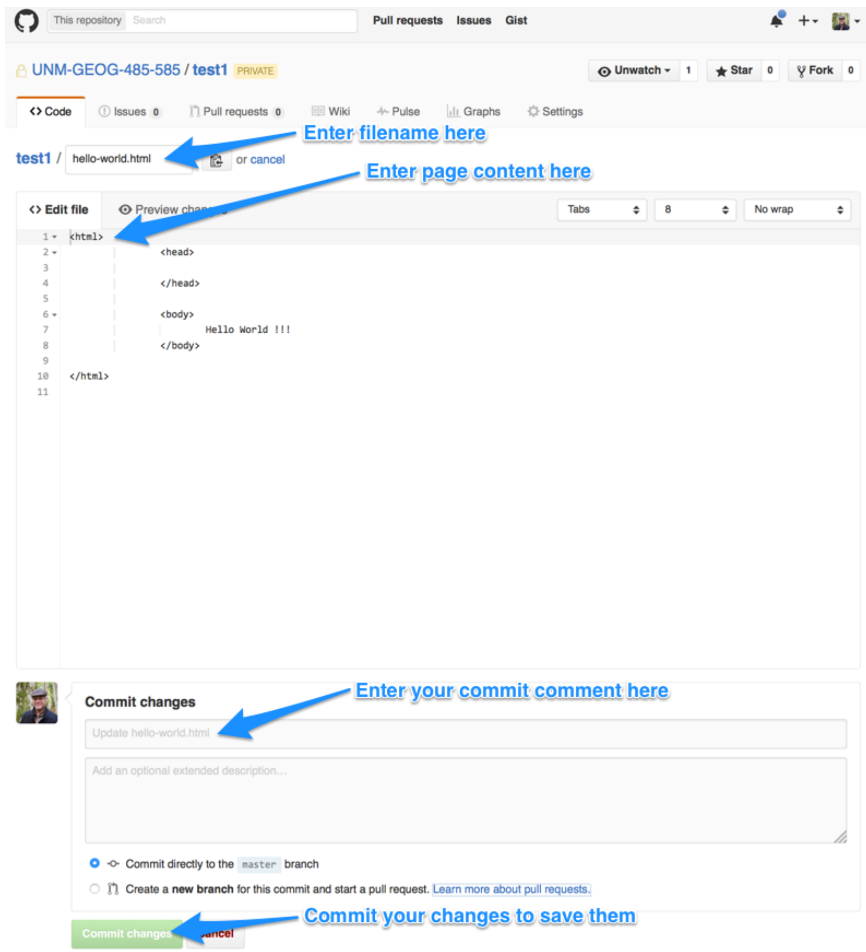

Figure 2.1: GitHub file creation/editor page

6. Keep the default option to “Commit directly to the master branch”
7. Click the “Commit New File” button to commit your change and save the file

Step 3 - Preview Your Web Page in a Browser

While we will later discuss strategies for hosting your web content on a system (GitHub for example) that supports direct access by web clients, to preview the web page you just created you need to download the repository to your local computer where you can open the locally stored file in a browser.

1. Go to the class organization page ((https://github.com/UNM-GEOG-485-585 - logging in if necessary) and click on your repository name in the list.
2. On the page that comes up listing the files in your repository, click the “Download Zip” button above the list of files. You may be prompted to provide a download location - if not you will need to find the default download location. Often it is the “Downloads” folder in your home directory.
3. Extract the contents of the downloaded .zip file using the appropriate utility program on your computer. On Macs and Windows computers this functionality is available through right-clicking on the file name in their respective file browsers.
4. Once you have extracted the contents of the zip file open the `hello-world.html` file that you created in a web browser - typically if you double-click on the file it will open in your default browser. You can also open it from within your browser of choice by using the “Open File” (or similar) option in the browser’s file menu.
5. Confirm that the display resembles something like the following:

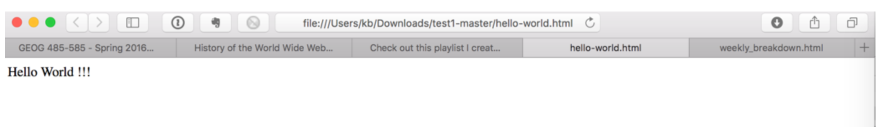

Figure 2.2: Sample `hello-world.html` file when viewed in a web browser

6. If the page does not appear as you like, edit it on GitHub and repeat 2-5 above until you see something like the sample figure.

Chapter 3

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

This week we will begin to build our foundation for developing material to be shared over the Internet via the World Wide Web. In particular we will cover the basic process of web development, define the parts of a web page, and spend some time learning about the different *languages* and define the key components of a web page: its structure, presentation, and behavior.

The presentation of information over the Internet is dependent upon the use of standards that have been developed for defining the *structure*, *presentation*, and *behavior* of content. This week we will begin working with the key technologies that define these three components of web content.

These concepts will be illustrated through reference to several simple web pages which are progressively modified to integrate all three of these components.

Expected Outcomes

By the end of this class module you should understand the following:

- The basic process of web development
- The parts of a web page
- The role of the three web page components: *structure*, *presentation*, and *behavior*
- Be able to write your own basic web page with your own content and make it available over the web

Key Concepts

- Parts of a web page
- Structure = X/HTML
- Presentation = CSS
- Behavior = Javascript
- Iterative Development

Class Prep

- Lynda.com tutorials
  - *Web Design Fundamentals*
    - 3. Getting Started
    - 4. Exploring Tools
  - *Version Control for Everyone*
    - 5. Basic Project Sharing
    - Conclusion
- Duckett, Jon, and Larsen, Rob. *Beginning HTML and CSS*. Somerset, NJ, USA: John Wiley & Sons, 2013. ProQuest ebrary. Web. 28 December 2015. This book is available online through the University Library - Chapters 1, 7, 10

Reference Materials

- Duckett, Jon, and Larsen, Rob. *Beginning HTML and CSS*. Somerset, NJ, USA: John Wiley & Sons, 2013. ProQuest ebrary. Web. 28 December 2015. This book is available online through the University Library - Chapters 2,3,4 and 8
- Lynda.com tutorials
  - *CSS Fundamentals*
  - *Javascript for Web Designers*

Weekly Milestone - Create a More Complex Web Page and Style It

This week's milestone activity takes you through the process of creating two more web pages in preparation for next week's work with the Google Maps API in developing your first web mapping page. These pages will be:

1. A *home page* for your portfolio that will be the access point for all of the materials you create (template/preview), and
2. Your first web page containing materials related to this *milestone* assignment (template/preview).

Step 1 - Open the *home page* template linked above in your web browser and open the preview in a second tab or window so that you can view both at the same time.

Step 2 - Copy the code in the home page template into a new text file named `index.html` on your computer.

Step 3 - Open the *milestone* assignment template linked above.

Step 4 - Copy the code in the template into a new text file named `milestone_02.html` on your copmuter.

Step 5 - After you have saved the `index.html` and `milestone_02.html` files to your hard drive open them up in your browser to see what they look like when read through a web browser.

Step 6 - Add your responses to the following questions to the `milestone_02.html` document. - note: it is a good practice when you are developing a web page to make small changes, save them, and preview the page to make sure that you have not made an error in your code before adding the next item. Practice this by adding each answer, saving your page and previewing it and correcting any errors in your code before going onto the next question.

Question 1 From examining the display of `index.html` in your web browser and the structure of the source code in the page, what effect (if any) does the white space (i.e. tabs, blank lines, multiple spaces) have on what is displayed in the browser?

Question 2 How are the

```
<h1>
```

and

```
<h2>
```

elements from the source code displayed differently in the browser?

Question 3 What type of element would you use to create additional list elements in either the "topic" or "data type" (`<ul>`) lists on the page.

Step 7 - Flesh out the `index.html` page that you created above (*Step 2*) with information specific to you based upon the content areas in the page. After making sure that your `index.html` and `milestone_02.html` are in the same directory, add a *relative* link to your `milestone_02.html` file to the "milestones" section of your `index.html` page by modifying the line

```
<p><a href="">Milestone 2</a></p>
```

to look like this

```
<p><a href="milestone_02.html">Milestone 2</a></p>
```

Save your change and test it in the browser by clicking the link on your `index.html` page in the browser. If it successfully opens your `milestone_02.html` page you have properly built your link.

Step 8 - Copy your `hello-world.html` file from *Milestone 1* into the same directory as your `index.html` file and modify the existing line in your `index.html` file

```
<p><a href="">Hello World</a></p>
```

to link to your `hello-world.html` file (follow the same pattern you used in *Step 7* above).

Step 8 - Make a copy of your `index.html` page by copying the content of the page and pasting it into a new document named `index_styled.html`.

Experiment with some of the styling capabilities described in Dave Raggett's "Adding a Touch of Style" page (http://www.w3.org/MarkUp/Guide/Style.html) on `index_styled.html` page you created above. Make at least three stylistic changes to the `index_styled.html` page. Add a link to your `index_styled.html` page to your home page (`index.html`) under the `milestones` section.

Step 9 - Transfer your created files `index.html`, `milestone_02.html`, and `index_styled.html` to your GitHub repository (created in *Milestone 1*). Of course you could do this by copying and pasting the content of your files into corresponding files in GitHub (but that would not be very efficient or satisfying), but you should probably experiment with SourceTree as demonstrated in this week's Lynda.com video tutorial as a way to work locally and transfer your files to GitHub for remote access and sharing.

Chapter 4

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

This week we will begin our work with the popular Google Maps *Application Programming Interface* (API) in developing an interactive web-based mapping client. This development activity will build upon the the work you've done over the last couple of weeks in developing basic web pages by using the capabilities that Google has made available for building mapping interfaces based upon their Maps platform. You will begin working with javascript as a client programming language to both interact with Google's servers and to provide the needed information for Google's mapping tool in your web page.

Expected Outcomes

By the end of this class module you should understand the following:

- What an Application Programming Interface (API) is
- How Javascript can be used to define the behavior of elements in a web page
- What the basic structure of a javascript code block for defining a Google Maps - enabled page looks like
- How to write a basic web page that includes an interactive Google Map

Key Concepts

- Application Programming Interface (API)
- Javascript and its location within an HTML page
- The interaction between javascript behaviors and structural elements in a web page

Class Prep

- Lynda.com tutorials
  - *Javascript for Web Designers* (included as a reference source last week)
    - 5. Using the Google Maps API

- Svennerberg, Gabriel. *Beginning Google Maps API 3*. Apress, © 2010. Books24x7. Web. Dec. 28, 2015. *Books 24x7 Library Database* - if this direct link to the book doesn't work for you, try logging in first and searching for `Google Maps API` - the Svennerberg book will be the first item on the list. 1-3 (skim chapter 2)

Continue reviewing:

- Duckett, Jon, and Larsen, Rob. *Beginning HTML and CSS*. Somerset, NJ, USA: John Wiley & Sons, 2013. ProQuest ebrary. Web. 28 December 2015. This book is available online through the University Library - Chapters 1, 7, 10

Reference Materials

- Duckett, Jon, and Larsen, Rob. *Beginning HTML and CSS*. Somerset, NJ, USA: John Wiley & Sons, 2013. ProQuest ebrary. Web. 28 December 2015. This book is available online through the University Library - Chapters 2,3,4 and 8
- Google Maps API Tutorial

Weekly Milestone - Creation of a Web Page with an Embedded Google Map

In preparation for creating a web page with an embedded Google Map you should first answer the following questions about what and how you want to map. As you define the type of map you want to build, think about a specific problem or topic that you would like to address with your map.

In this exercise you will be generating the configuration for the base map (i.e. The Google Maps background layers). In future assignments you will add your own custom content to free-standing web pages that include a mapper based upon the base map you define here.

Create a web page (based upon the assignment template) that contains your milestone writeup (including the embedded Google Map required by question 5), and link it to the home page (`index.html`) file you created last week.

Respond to Question 1-4 with an understanding that you are generating a web page that is designed for public viewing (even if you don't choose to make it public at this time), and should be both clear and complete.

Question 1 What area do you want to depict in your map? Why?
Question 2 What is the center point (latitude and longitude) of your area of interest?
Question 3 What style of map (roads, satellite, hybrid, terrain) is appropriate for your map? Why?
Question 4 What is the scale of your map (local, regional, continental, global)? How will this translate into your selection of an appropriate default zoom level for your map?

Now that you have answered these questions about the map that you want to create, refer to the examples in the lecture notes, the Google Maps Tutorial, and this week's reading (link to the code for Svennerberg's Chapter 3 example) and video tutorial assignment to create a custom Google map.

Question 5 Embed a Google Map in your writeup that is based upon your responses to questions 1-4 above.

Chapter 5

Week 4 - Module 2a - Web-based Mapping Clients. Google Maps API

This week covers some additional topics related to the Google Maps API, particularly focusing on styling the Maps base maps using the styled maps wizard and integrating the javascript generated by the wizard into the base web page code developed last week; and, using Google's Fusion Tables tool to create and manage tabular data for mapping and other visualization. We complete our work with the Maps API with an example of a more "real" example of a maps-enabled web page.

Expected Outcomes

By the end of this class module you should be able to:

- Generate a Google Maps JSON style using the *Styled Maps Wizard*
- Integrate that JSON into your map client page for styled basemap display

You should also understand

- The potential of *Fusion Tables* as an alternative source of data to integrate into a custom Google Map page
- The potential structure of an *operational* web page, including the physical separation of page components (structure, presentation, behavior) into separate files

Key Concepts

- Generating Google Maps styles
- Integrating styles into a Google Maps page
- Fusion tables as a data source for Google Maps maps
- Separation of structure, presentation, and behavior in web development

Class Prep

- Lynda.com tutorials
  - *Javascript for Web Designers* (continued)
    - 5. Using the Google Maps API

Reference Materials

- Duckett, Jon, and Larsen, Rob. *Beginning HTML and CSS*. Somerset, NJ, USA: John Wiley & Sons, 2013. ProQuest ebrary. Web. 28 December 2015. This book is available online through the University Library - Chapters 2,3,4 and 8
- Google Maps API Tutorial
- Google Maps Styling Reference
- Svennerberg, Gabriel. *Beginning Google Maps API 3*. Apress, © 2010. Books24x7. Web. Dec. 28, 2015. *Books 24x7 Library Database* - if this direct link to the book doesn't work for you, try logging in first and searching for `Google Maps API` - the Svennerberg book will be the first item on the list. 4-8
- Google Maps Fusion Mapper

Weekly Milestone - Styling of an Embedded Google Map

Make a free-standing web page based upon the Google Map that you created as part of last week's lab assignment. Use the Google styled maps wizard to define *at least* three modified base map styles and integrate the JSON generated by the wizard into your new Google Map page.

Deep Dive - Creation of a a Google Maps Web Page with Custom Points and Labels

In your milestone for Week 4 you built a styled Google Maps base map for a particular region of interest. For this *deep dive* assignment create a new free-standing web page that includes a brief description of the topical focus of your mapper:

- The type of information that you want to depict in your map
- Your reasons for selecting the specific area shown in the map
- A description of what you are trying to communicate with the map

Embed the base map that you initially created for your milestone into this new web page.

- Add 5 overlay objects to the map that relate to specific items of interest or importance. These overlay objects may be *markers*, *polylines*, or *polygons*. Make sure to include descriptive titles for each object.
- Add an *infobox* to each object that contains additional detailed information about the object

Chapter 6

Week 5 - Module 3 - GIS and Services Oriented Architectures

Core the the development of distributed mapping systems over the internet is the concept of web services and the interoperability upon which they are based as the means of communication between systems. This week's lecture and focuses on the core concepts of geospatial *Services Oriented Architectures* and the open interoperability standards from the *Open Geospatial Consortium* that enable the exchange of map images and data over the web.

Expected Outcomes

By the end of this class module you should understand the following:

- The difference between raster and vector data formats and strategies for retrieving information about supported file formats
- The three general tiers of a geospatial services oriented architecture and the components that may exist in those tiers
- The key Open Geospatial Consortium standards for access, data, and representation

Key Concepts

- Raster and Vector Data Models
- The tiers of a geospatial services oriented architecture
- The constituent components of SOA tiers
- The role of OGC services in providing connectivity between SOA tiers
- The OGC WMS, WFS, WCS, GML, and KML standards and their respective capabilities and purposes

Class Prep

1. Yang C, Raskin R, Goodchild M, Gahegan M. Geospatial Cyberinfrastructure: Past, present and future. Computers, Environment and Urban Systems. 2010;34: 264–277. doi:10.1016/j.compenvurbsys.2010.04.001 http://www.sciencedirect.com.libproxy.unm.edu/science/article/pii/S0198971510000268

2. Granell C, Díaz L, Gould M. Service-oriented applications for environmental models: Reusable geospatial services. Environmental Modelling & Software. 2010;25: 182–198. doi:10.1016/j.envsoft.2009.08.005 http://www.sciencedirect.com.libproxy.unm.edu/science/article/pii/S1364815209002047

3. Foster I. Service-Oriented Science. Science. 2005;308: 814–817. doi:10.1126/science.1110411 http://science.sciencemag.org.libproxy.unm.edu/content/308/5723/814

Reference Materials

None

Weekly Milestone - Fun with data

Question 1 Define a data theme that you would like to focus on for this assignment

Download three data products from one or more of the following online data repositories or another data repository that has data that interest you.

- New Mexico Resource Geographic Information System
- The US National Map Data Download Site
- NOAA's National Climate Data Center *Climate data online: Data discovery* site
- US Census Bureau - Geography - TIGER Data

Question 2 For each of the three datasets provide the following information

- The name of the dataset
- The filename(s) for the dataset
- A short (1-2 sentence) description of the dataset's contents
- The bounding box (provided as the minimum and maximum extent in the N-S and E-W directions) in the native units and coordinate system
- The coordinate reference system - by name and EPSG code

Chapter 7

Week 6 - Module 4.1 - Interoperability Standards - XML, KML, and WMS

This week's class focuses on three open interoperability standards that are the most broadly used of the standards that we will be covering.

Extensible Markup Language (XML) The World Wide Web Consortium (W3C) standard that is the foundation for many other service and data standards including: the service metadata (GetCapabilities) for the OGC WMS, WFS, and WCS, Geography Markup Language (GML), and KML.

KML Formerly known as Keyhole Markup Language, an OGC standard since 2008, KML is a combined geospatial data and representation standard that enables the combined transfer of both location-based data and styling information within a defined XML model.

Web Map Service (WMS) The OGC standard for providing on-demand map visualizations based upon user provided parameters reflecting selected data layers, defined areas of interest, image formats, and optionally time of interest.

Expected Outcomes

At the end of this class students should have an understanding of the following:

- The basic characteristics of XML documents, including the concepts of *well-formed* and *valid* XML
- The capabilities of KML for providing both data and representation information for geospatially referenced data.
- The request-response model for OGC WMS, including the required and optional request parameters for the *GetCapabilities*, *GetMap*, and *GetFeatureInfo* requests; and the response types generated in response to those requests.
- A general familiarity with the linkage between the WMS and KML standards

Key Concepts

- XML as a general standard for structured data exchange, with DTDs and Schemas defining application specific data models
- KML as a data and representation standard for delivery of geospatial data and symbolization information into client applications, both desktop and web-based.
- WMS as a geospatial data visualization standard for providing online access to map images in a variety of formats for integration into desktop and web-based mapping applications

Class Prep

- OGC Workshop White Paper

Reference Materials

- OGC WMS Implementation Specification Version 1.0 - 2000, Version 1.1 - 2001, Version 1.1.1 - 2002, *Version 1.3.0 - 2006*
- OGC KML Version 2.2 - 2008, *Version 2.3 - 2015*
- Google Code KML Documentation

Weekly Milestone - WMS & KML

There are a large number of WMS services available on the web. One way to find interesting services is to search for them using standard search engines such as Google. Try searching for the following search phrase:

`"REQUEST=GetCapabilities" and "SERVICE=WMS"`

as a single search phrase

Question 1 What search engine did you use?
Question 2 How many 'hits' did you get?
Question 3 How useful (generally in terms of getting a pointers to live WMS services [defined as a *functioning* GetCapabilities request]) were the 'hits'?

Pick two of the services that included live "GetCapabilities" requests that you found above, and answer the following questions about each.

Question 4 (service #1) What is the URL for the full GetCapabilities request to the service?

What is the Name of the service?

What Format(s) are available for GetMap requests from the service?

How many layers are included in the service (*including nesting layers*)?

Question 4 (service #2) What is the URL for the full GetCapabilities request to the service?

What is the Name of the service?

What Format(s) are available for GetMap requests from the service?

How many layers are included in the service (*including nesting layers*)?

Question 5: For one of the layers in the first service, What is the name of the layer?

What is the SRS of the layer?

What is the name of the projection that matches the SRS EPSG code?

What is the LatLonBoundingBox of the layer?

Open the following GetCapabilities request in your browser. Select "View Source" from the browser menu to see the delivered XML document (it may appear as an unformatted string of text by default in your browser - if that is the case, save the file to your hard drive and view it in a text editor). Use the information in the XML capabilities document to formulate `GetMap` requests for the following map images. Include the requests and resulting images in your write-up. Comment on anything unusual that you notice in the images that are returned.

http://gstore.unm.edu/apps/rgis/datasets/92403ebf-aec5-404b-ae8a-6db41f388737/services/ogc/wms?SERVICE=wms&REQUEST=GetCapabilities&VERSION=1.1.1

Question 6 for the area surrounding Bernalillo County (-107.2,34.7,-106,35.25; EPSG:4326) for the `g_2007fe_35_county` layer as a 200x200 pixel JPEG

for the same area and layer as a 500x500 pixel PNG

Open the following (linked) KML file in Google Earth, uncompress it, and save the contained KML file on your computer. Open the KML file in a text editor (e.g. Text Wrangler [Mac], Notepad/Notepad++ [Windows]).

http://rgis.unm.edu/gstore/datasets/3f0a85aa-b7f8-47bd-8db6-1c0e66becf72/nm__state__bdy__00.derived.kml

Question 7 Add a second *Placemark* element to the KML file that represents a *square* region that is completely contained within the state boundary. Save the KML file and open it in Google Earth (download from http://www.google.com/earth/index.html). Submit the KML file (as a link in your writeup) as part of your writeup for the milestone.

Deep Dive - OGC Service Concepts

Question 1 What request type is common across all three (WMS, WFS, WCS) OGC web services that we have learned about?

Answer the following questions about a *WMS GetCapabilities* request

Question 2 What are the required parameters, and what do they represent?

What is returned in response to a WMS GetCapabilities request?

Answer the following questions about a *WMS GetMap* request

Question 3 What are the required parameters, and what do they represent?

What is returned in response to a WMS GetMap request?

What is the significance of transparency in WMS requests?

Question 4 What OGC request would you use to inform the configuration of a client application (like ArcGIS or QGIS) about an OGC service that you want to add layers from?

Which OGC request would you submit under the following circumstances (*include both the service type* [e.g. WMS, WFS, WCS], and the *request* [e.g. GetMap, GetCapabilities, GetCoverage, etc.] in your answer)

Question 5 You want a map image representing three layers of data in a single JPEG for a specified area of interest.

You want to retrieve data representing geometries and associated attributes for a road network, with the returned data in GML.

You want to retrieve data representing a digital elevation model (a raster dataset) in the form of a GeoTIFF.

Question 6 - What are the EPSG codes of the following Spatial Reference Systems WGS 84 (Geodetic CRS [geographic 2d])

NAD83 / UTM zone 13N

NAD27 / UTM zone 13N

Retrieve the GetCapabilities XML response from the following WMS, and answer the following questions.

http://gstore.unm.edu/apps/rgis/datasets/715663ba-c1c3-414c-84a7-c671526f8316/services/ogc/wms?SERVICE=wms&REQUEST=GetCapabilities&VERSION=1.1.1

Question 7 What is the Title of the service?

Who is the Contact Person for questions about the service?

What are the available image formats for the GetMap request for this service?

What are the SRS/CRS's for which layers from this service are available (remember that nested layers inherit the SRS/CRS of their parent layers).

Question 8 Formulate a GetMap request for the "tl_2010_35_bg10" layer from this service, for a 500x500 pixel map image that is 0.05-degrees wide and 0.05-degress high, with the SW corner of the map image located at 35°N and -106°45'E (EPSG:4326). Include in your write-up the complete GetMap request and the returned map image.

Chapter 8

Week 7 - Module 4.2 - Interoperability Standards - WFS & WCS

This week's class focuses on two other key Open Geospatial Consortium standards that were created to enable access to geospatial *data* of a variety of types.

Web Feature Service (WFS) A standard designed for providing on-demand access to features (typically points, lines, and polygons and more complex combinations of these feature types) and their associated attributes, in a variety of formats, and optionally filtered by spatial and other query parameters.

Web Coverage Services (WCS) A standard focused on providing access to *coverages* representing a variety of data types, but particularly optimized for dynamic delivery of data based upon multi-dimensional gridded data.

Expected Outcomes

At the end of this class students should have an understanding of the following:

- The request-response model for OGC WFS, including the required and optional request parameters for the *GetCapabilities*, *DescribeFeatureType*, and *GetFeature* requests; and the response types generated in response to those requests.
- The request-response model for OGC WCS, including the required and optional request parameters for the *GetCapabilities*, *DescribeCoverage*, and *GetCoverage* requests; and the response types generated in response to those requests.
- An understanding of the distinction between WMS, WFS, and WCS for use in different usage scenarios (e.g. map images, vector, raster data access)

Key Concepts

- OGC WFS as a data access standard for *features* and their attributes with support for a variety of query methods including spatial and parameter values.
- OGC WCS as a data access standard for *coverages* representing spatio-temporal gridded data represented in 1- or more dimensions.

Class Prep

Reference Materials

- [OGC WFS Implementation Specification][http://www.opengeospatial.org/standards/wfs]
- [OGC WCS Implementation Specification][http://www.opengeospatial.org/standards/wcs]

Weekly Milestone - WMS GetMap Requests, Map Scale and Aspect Ratio Calculations

You might have noticed in the WMS requests that you generated in the previous lab returned images that didn't look "quite right" relative to what you may know of the shape of familiar features.

For example, a WMS request for a 200x200 pixel PNG file (Figure 8.1 for an area surrounding Bernalillo County (-107.2,34.7,-106,35.25) from the previous lab would be (link):

```
http://gstore.unm.edu/apps/rgis/datasets/92403ebf-aec5-404b-ae8a-6db41f388737/
services/ogc/wms?VERSION=1.1.1&SERVICE=WMS&REQUEST=GetMap&BBOX=-107.2,34.7,
-106,35.25&LAYERS=g_2007fe_35_county&FORMAT=image/png&TRANSPARENT=TRUE&
STYLES=&SRS=EPSG:4326&WIDTH=200&HEIGHT=200
```

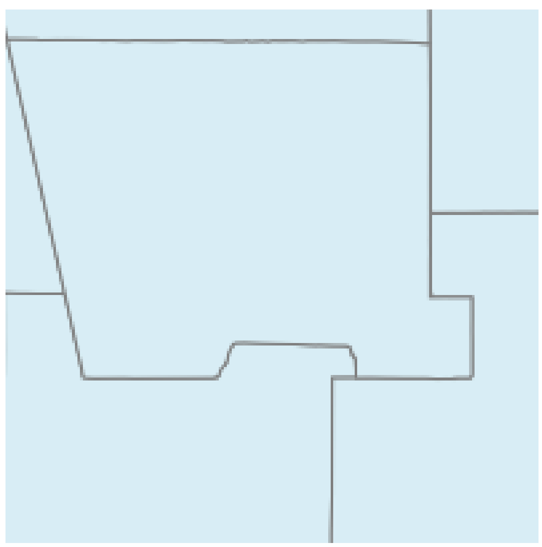

Figure 8.1: Returned map image for the region surrounding Bernalillo County for a WMS request with `BBOX=-107.2,34.7,-106,35.25`, `WIDTH=200` and `HEIGHT=200`

this request results in a map image that does not agree with the standard shape of Bernalillo county (depicted in QGIS - Figure 8.2) that we are accustomed to, regardless of the specific map projection being used.

This discrepancy is the result of a difference in the aspect ratio of the requested BBOX (-107.2,34.7,-106,35.25) and the requested image dimensions (200x200 pixels). *When you compose a WMS GetMap request, you need to make sure that the aspect ratio of both the image size and BBOX match.*

For example, if we calculate the aspect ratio of the BBOX we obtain the following values (remember that the BBOX is specified as a comma separated list of x,y coordinates: minx,miny,maxx,maxy):

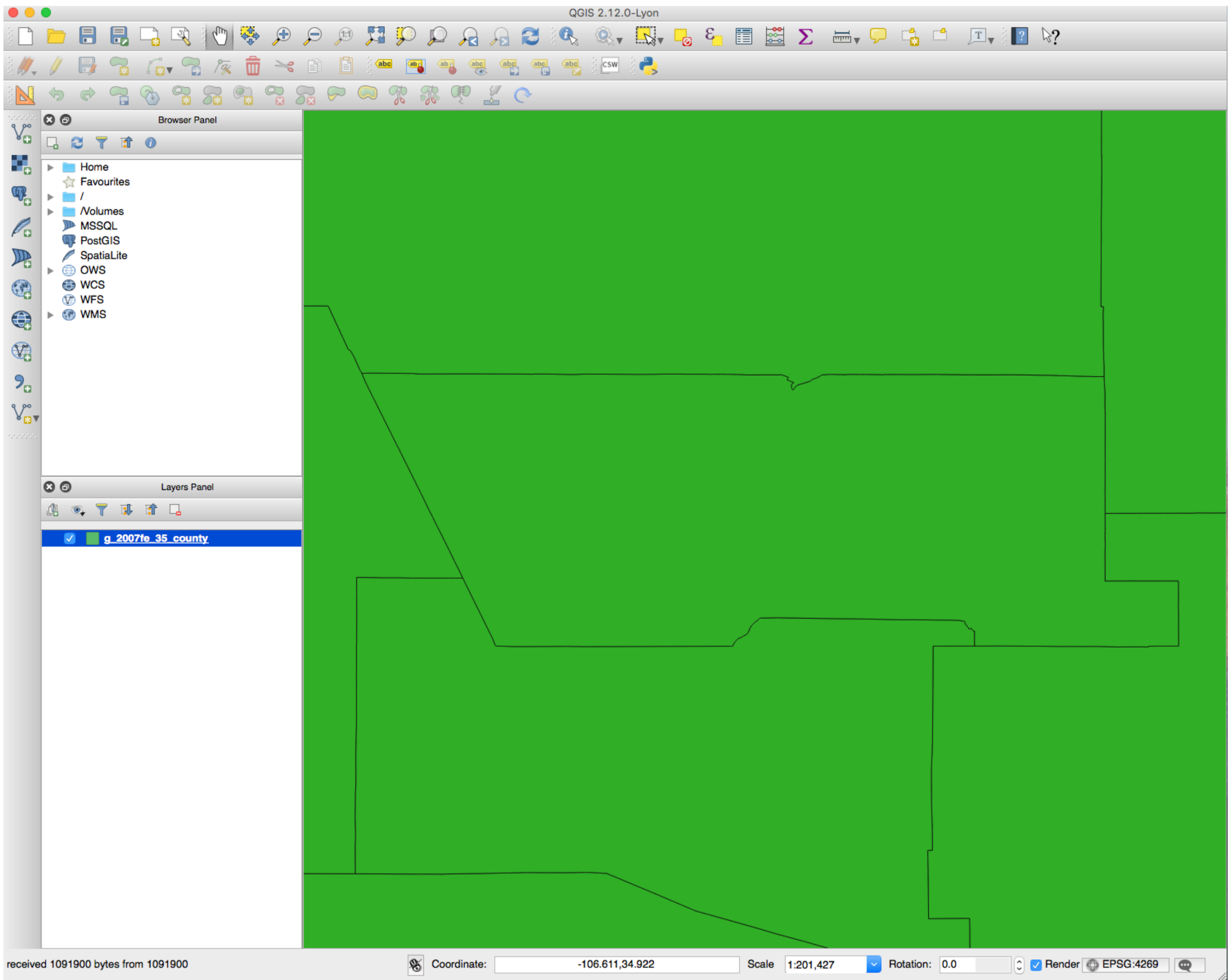

Figure 8.2: Area around Bernalillo County as viewed in QGIS based on the OGC WCS based on the same data source used for the WMS request illustrated above}

$BBOX_{width} = x_{max} - x_{min} = (-106) - (-107.2) = 1.2°$

$BBOX_{height} = y_{max} - y_{min} = 35.25 - 34.7 = 0.55°$

$BBOX_{aspect-ratio} = BBOX_{width}/BBOX_{height} = 1.2°/0.55° = 2.1818$

If we want to retrieve a map image that is 200 pixels wide, we need to calculate an image height that yields an aspect ratio that matches the BBOX aspect ratio. Harking back to basic algebra:

$Image_{width} = 200px$

$Image_{aspect-ratio} = Image_{width}/Image_{height} = 200px/Image_{height} = 2.1818$

$Image_{height} = Image_{width}/aspect - ratio = 200px/2.1818 = 91.667px$

So, if we request an image that is 200x92 (we have to request pixel dimensions in integers, so rounding to the nearest integer) we should get a representation that closely approximates the proper shape of features. The modified WMS request with the new image's size is the following (link):

```
http://gstore.unm.edu/apps/rgis/datasets/92403ebf-aec5-404b-ae8a-6db41f388737/
services/ogc/wms?VERSION=1.1.1&SERVICE=WMS&REQUEST=GetMap&BBOX=-107.2,34.7,-106,35.25&
LAYERS=g_2007fe_35_county&FORMAT=image/png&TRANSPARENT=TRUE&STYLES=&
SRS=EPSG:4326&WIDTH=200&HEIGHT=92
```

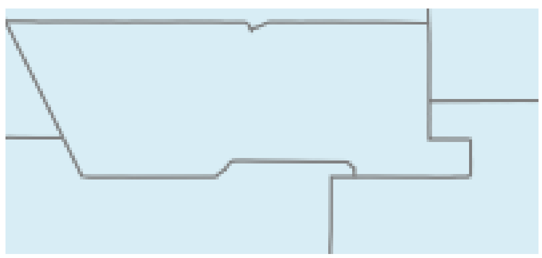

Figure 8.3: Returned map image for the region surrounding Bernalillo County for a WMS request with `BBOX=-107.2,34.7,-106,35.25`, `WIDTH=200` and `HEIGHT=92`

This process may be reversed to request images of a fixed size for use in a client interface, with the requested BBOX modified to match the aspect ratio of the target image. If, for example, images are being requested for a client interface with a fixed map size of 600x400 pixels, a corresponding BBOX can be derived using the same calculation.

If, for example, the area of interest for a map is 2 degrees wide, we can calculate the target height (in degrees) using the aspect ratio of the desired image.

$Image_{aspect-ratio} = Image_{width}/Image_{height} = (600px)/(400px) = 1.5$

$BBOX_{aspect-ratio} = BBOX_{width}/BBOX_{height} = 2°/BBOX_{height} = 1.5$

$BBOX_{height} = BBOX_{width}/BBOX_{aspect-ratio} = 2°/1.5 = 1.3333°$

If our area of interest extends from -106 to -108 degrees East Longitude, we can use the known target height of 1.3333 to generate a WMS BBOX of the appropriate aspect ratio. If the minimum Latitude of interest is 34.7 degrees North Latitude, the maximum BBOX Y value would be

$y_{max} = y_{min} + BBOX_{height} = 34.7° + 1.3333 = 36.0333$

This set of calculations may be used to compose the following WMS request (link):

```
http://gstore.unm.edu/apps/rgis/datasets/92403ebf-aec5-404b-ae8a-6db41f388737/
services/ogc/wms?VERSION=1.1.1&SERVICE=WMS&REQUEST=GetMap&BBOX=-108,34.7,-106,36.0333&
LAYERS=2007fe_35_county&FORMAT=image/png&TRANSPARENT=TRUE&STYLES=&
SRS=EPSG:4326&WIDTH=600&HEIGHT=400
```

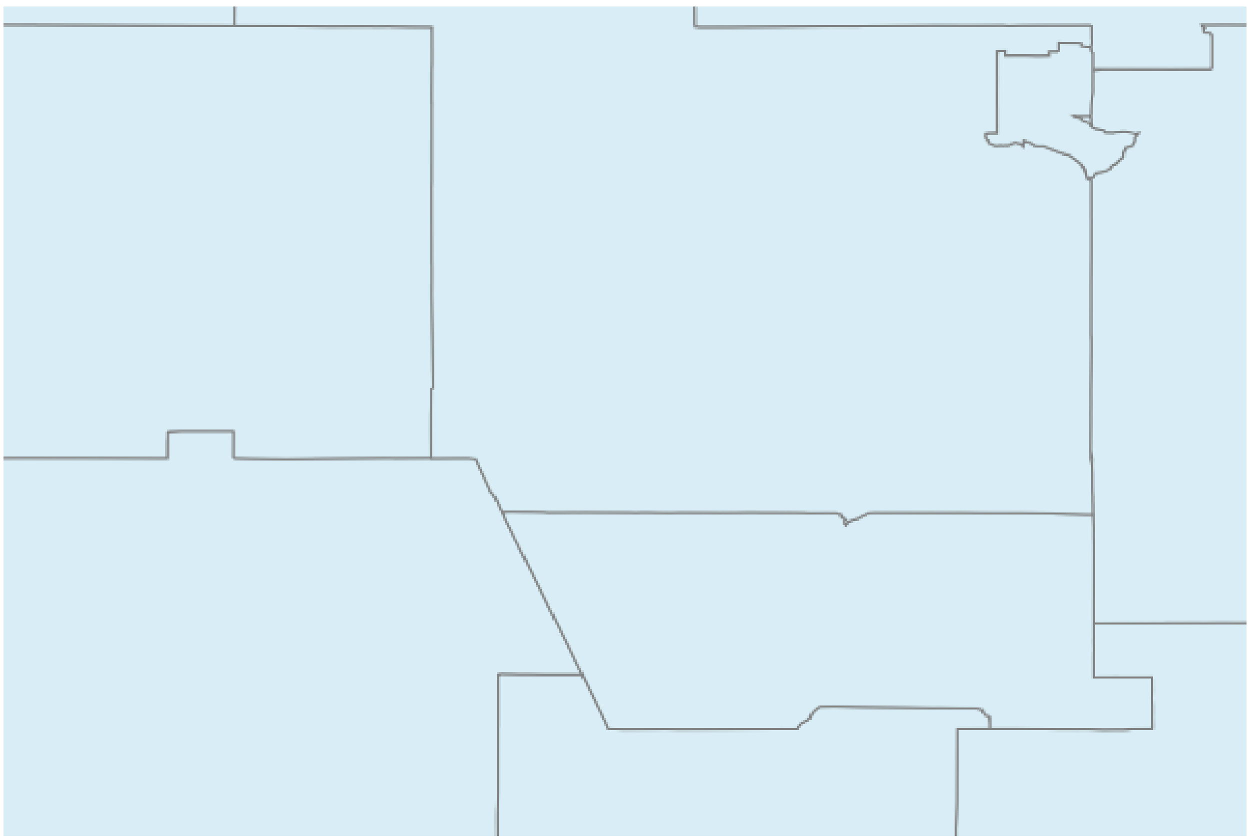

Figure 8.4: Returned map image for the region surrounding Bernalillo County for a WMS request with `BBOX=-108,34.7,-106,36.0333`, `WIDTH=600` and `HEIGHT=400`

Given that McKinley County NM is contained within the following BBOX: -109.5, 34.5, -106.5, 36.5

Question 1 What is the aspect ratio ($BBOX_{width}/BBOX_{height}$) of this geographic region?
Question 2 What would be the height (in whole pixels - round up) for a map image for this region that is 600 pixels wide?
Question 3 Formulate a WMS request that reflects the values determined in 1.1 and 1.2 above for the WMS service used above in the examples and for the `g_2007fe_35_county` layer. Include in your answer both the complete WMS request and the returned map image (either as a static image, or an image that is linked to the live WMS).
Question 4 Formulate a WMS request for a 900x600 pixel map image that represents the full 3-degree width of the geographic region, and is based upon the minimum Y value of 34.5 degrees North Latitude. Include in your answer both the WMS request and the returned map image (either as a static image, or an image that is linked to the live WMS).

Given the following set of `GetMap` requests against the *USGS/EROS Ortho Imagery Service* answer the following questions

1) http://raster.nationalmap.gov/arcgis/services/Orthoimagery/USGS_EROS_Ortho_SCALE/ImageServer/WMSServer?request=GetMap&service=WMS&VERSION=1.1.0&SRS=EPSG:4326&LAYERS=0&FORMAT=image/jpeg&TRANSPARENT=FALSE&STYLES=&WIDTH=1200&HEIGHT=800&BBOX=-107.1242070,34.7509960,-106.1242070,35.4176960

2) http://raster.nationalmap.gov/arcgis/services/Orthoimagery/USGS_EROS_Ortho_SCALE/ImageServer/WMSServer?request=GetMap&service=WMS&VERSION=1.1.0&SRS=EPSG:4326&LAYERS=0&FORMAT=image/jpeg&TRANSPARENT=FALSE&STYLES=&WIDTH=1200&HEIGHT=800&BBOX=-106.6867070,35.0426773,-106.5617070,35.1260148

3) http://raster.nationalmap.gov/arcgis/services/Orthoimagery/USGS_EROS_Ortho_SCALE/ImageServer/WMSServer?request=GetMap&service=WMS&VERSION=1.1.0&SRS=EPSG:4326&LAYERS=0&FORMAT=image/jpeg&TRANSPARENT=FALSE&STYLES=&WIDTH=1200&HEIGHT=800&BBOX=-106.6281133,35.0817417,-106.6203008,35.0869503

4) http://raster.nationalmap.gov/arcgis/services/Orthoimagery/USGS_EROS_Ortho_SCALE/ImageServer/WMSServer?request=GetMap&service=WMS&VERSION=1.1.0&SRS=EPSG:4326&LAYERS=0&FORMAT=image/jpeg&TRANSPARENT=FALSE&STYLES=&WIDTH=1200&HEIGHT=800&BBOX=-106.6246953,35.0840205,-106.6237187,35.0846715

Question 6 Which layer(s) return map images that display image content (i.e. return a non-blank image)?
Questions 7 What is the difference between these requests? How might this difference influence whether or not a map image with content is being returned?
