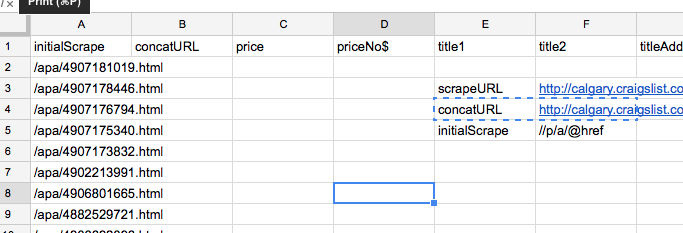
GeoDuo: Anna Chinn and Rachel Bloom

GEOG 407: Assignment 2

1. Web scrapes are a useful way of automating the retrieval of large amount of data. By using Xpath commands in Google spreadsheets, our team was able to identify and scrape information about apartment listings in Calgary from the Craigslist website. Our team was able to identify and extract addresses in order to geocode listing advertisements. Then we added the scraped data, including the geocoded addresses, to a KML and mapped them using the Google My Maps platform. The final result is a map that displays location, information about the rentals, and provides a link to the original Craigslist advertisement.

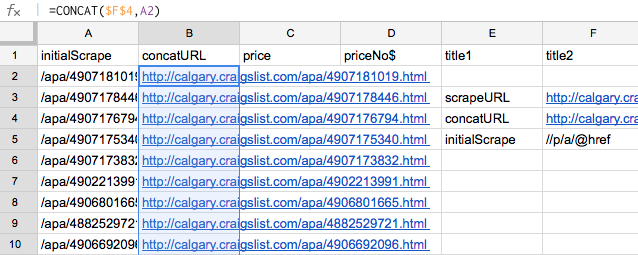
2.

We initially scrape listing URLs using the XPath command //p/a/@href in conjunction with the craigslist URL “<http://calgary.craigslist.ca/search/apa/>” (ie. =IMPORTXML(craigslistURL,XPath).

XPath command: //p/a/@href 

In column B, we acquired the direct links to the listings by using a CONCAT function to concatenate a generalized Craigslist URL prefix “<http://clagary.craigslist.ca>” with each scraped listing suffix from column.

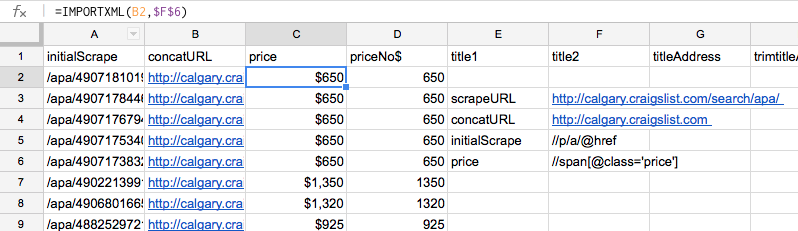
Spreadsheet function: =concat(concatScrape,initialScrape)



All following scrapings are based on nested queries that scrape these concatenated URLs

(ie. =IMPORTXML(ColumnBcell,XPath)

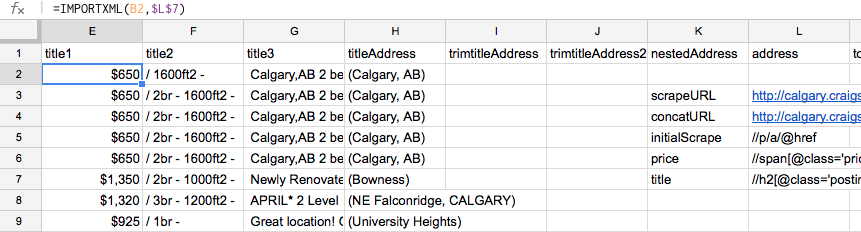
In order to scrape price listings, we looked at the page source and determined tags that encompassed price values.

Xpath command: //span[@class=’price’]).

The address was a combination of two scrapes, one of addresses listed in brackets in the title of the listing, and in a div tag called ‘mapaddress’.

Scraping the title from the listing using the XPath command //h2[@class=’postingtitle’] is automatically split into 4 columns in Google spreadsheet based the tags in the source code.

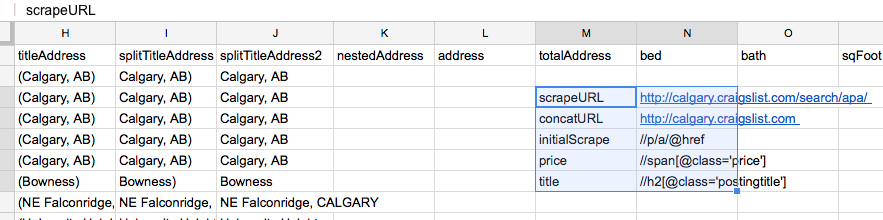
XPath command: //h2[@class=’postingtitle’]



The 4th column of the title a term between brackets that often contains some sort of location information. We called this titleAddress

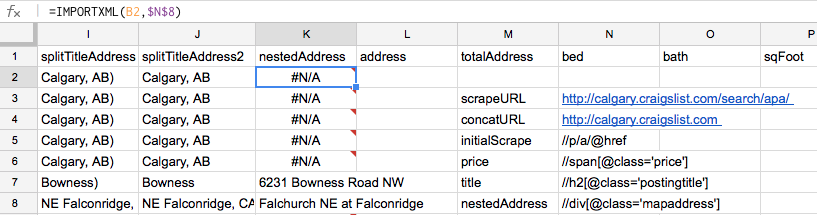
We trimmed the brackets of the titleAddress using split functions in Google spreadsheet.

Spreadsheet function: =SPLIT(titleAddress, “(“) and =SPLIT(splitTitleAddress, “)”)



We scraped a second piece of location data, as well and called it nestedAddress

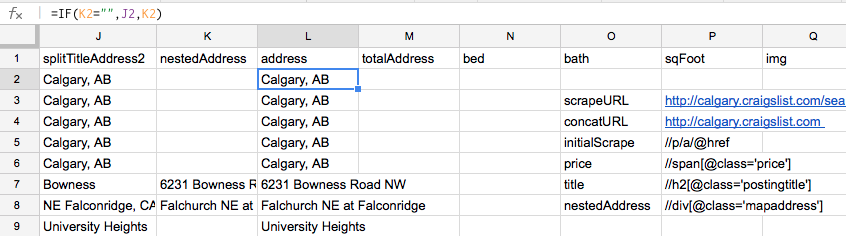
XPath command: //div[@class=’mapaddress’]



At this point, there were cells with missed location data that displayed either “#VALUE” or “#N/A”. To eliminate all these cells, we copy and pasted the values of this current sheet onto a new spreadsheet and found and replaced these dataless cells with nothing (ie. “”)

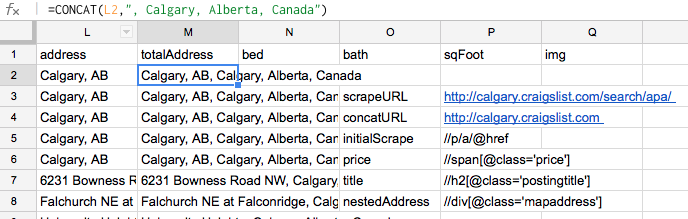
After inspecting the data, we figured that most of the time, the nestedAddress was more precisely located than the splitTitleAddress2 data. In a new column called address, we used a function that could grab the information in the nestedAddress column, unless nestedAddress was empty, in which case it would grab the location data from splitTitleAddress2.

Spreadsheet function: =IF(nestedAddress= “”, splitTitleAddress2, nestedAddress)

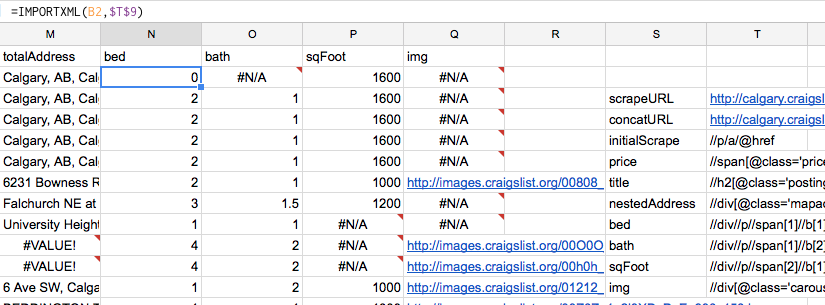


We then concatenated the resultant address column with the string “, Calgary, Alberta, Canada” to be sure the coordinates returned by the geocoder would be in Calgary, as opposed to in some other city with similar addresses. Additionally, listings with no location data whatsoever would at least be located in the city Calgary, probably at the city area centroid. We called this column totalAddress.

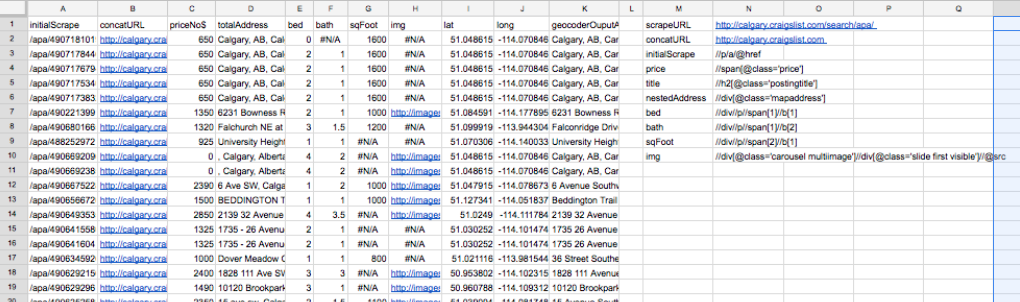
Spreadsheet function: =CONCAT(address,”, Calgary, Alberta, Canada)

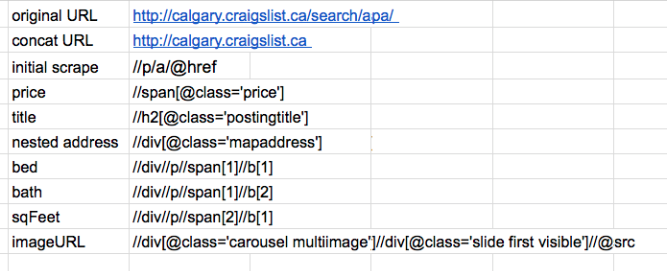


We then added other XPath commands to import number of bedrooms/bathrooms, the square footage, and the first image on the listing

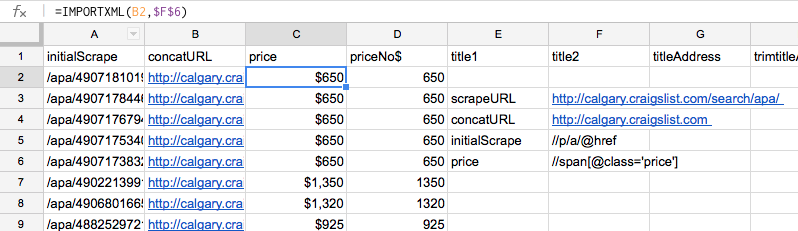


Clean spreadsheet with XPath Commands:

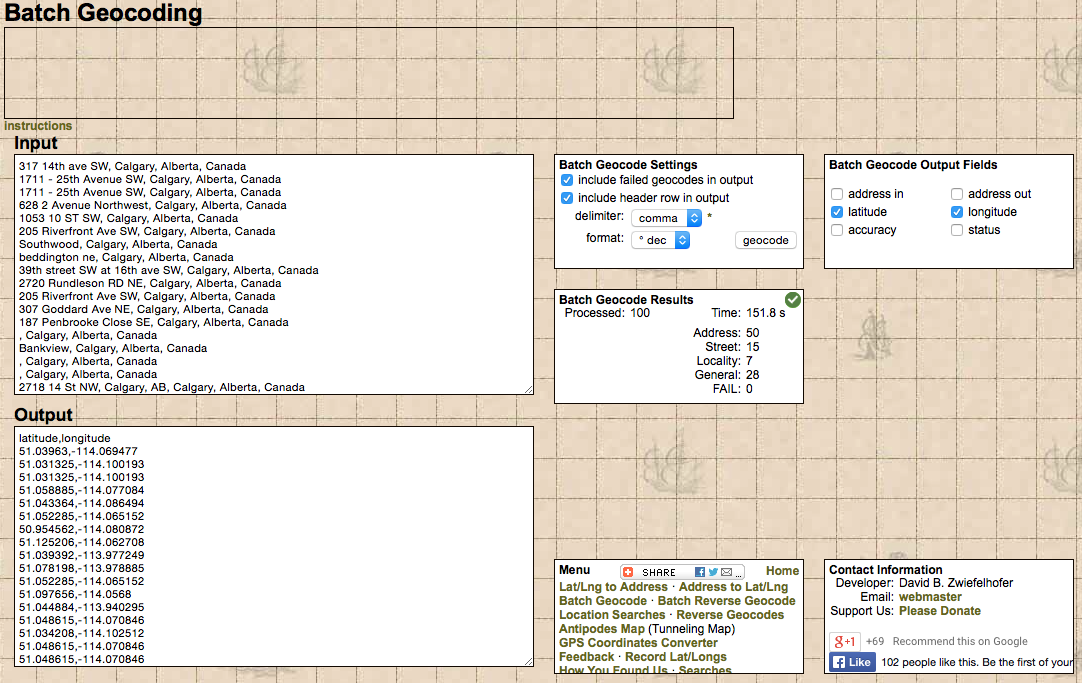




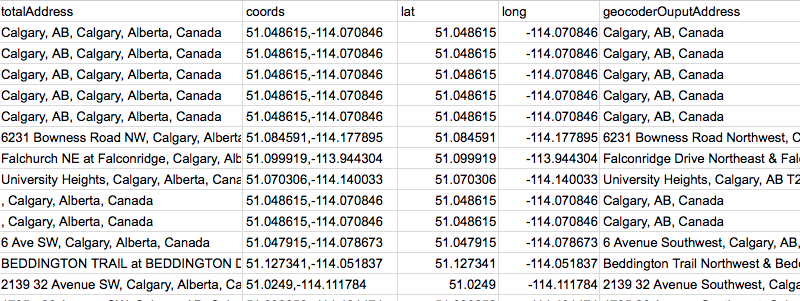
3. We used the XPath command //span[@class=’price’] to scrape the price off from each listing’s concatenated URL (column B). We then copied the price values into a new column and formatted the cells as numbers, as opposed to currency.



4.a.We copy and pasted the totalAddress column into this batch geocoder: <http://www.findlatitudeandlongitude.com/batch-geocode/>



The geocoder returns coordinates separated by commas in same order the addresses were input into the geocoder. The coordinates could easily be copy and pasted back into a new row of the spreadsheet and split into separate latitude and longitude columns using a =SPLIT(coordinates,”,”) function.

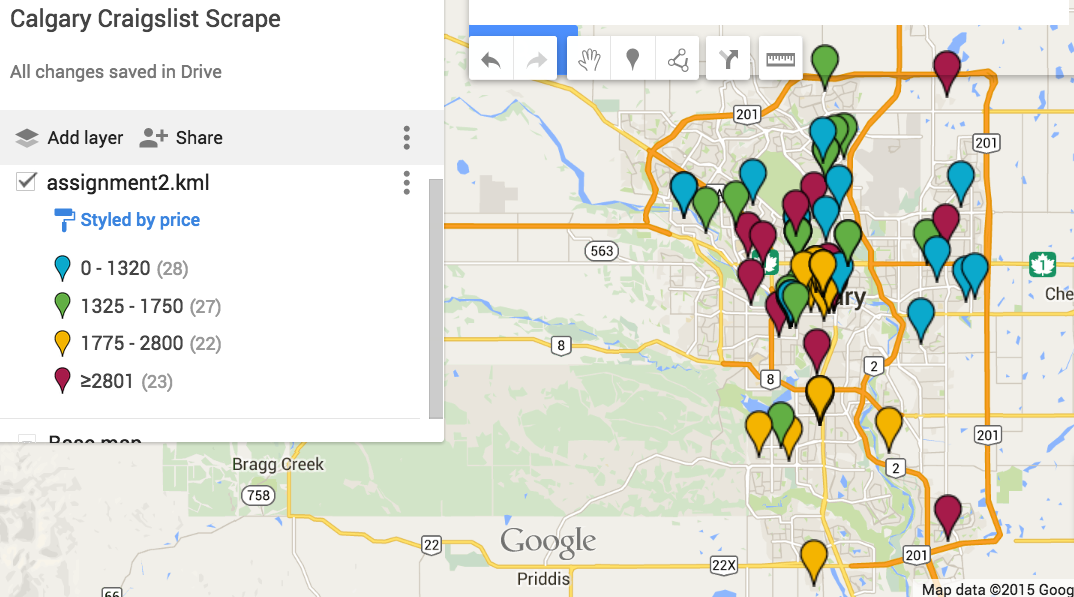


b/c. In order to create the .KML file, we programmed a “for” loop in Python (because Dipto said that Javascript cannot easily manipulate .CSV files) that would read the .CSV that we generated from our clean spreadsheet. The loop printed out a .KML formatted placemark for each row of scraped data. If/else statements to vary the placemark style name based on price value so that we could vary the colour of the placemarks. We then copy and pasted the print statement containing all our placemarks into Notepad++ and added the proper <kml> tags. We hardcoded the styles that would vary the Placemark colour. The code we used to automate the data from the .CSV to create a .KML file is as follows:



d. As it turns out, the Google MyMaps allows you to change the placemark colour as part of its interface. Despite this, with the placemark colours intrinsic to the code, the .KML can be opened in other programs, like Google Earth and still display colourful placemarks.

This is the .KML Uploaded in Google MyMaps:



5. As a note on automation, all the spreadsheet functions listed were applied to all cells by using the drag copying functionality of the program and using $ to maintain cell values when necessary

6.

Anna

a. Actually dealing with Google spreadsheet was the hardest part of the process. Often times, the spreadsheet would not load the scraping in certain cells, though the XPath command was fine. We had to restart our entire spreadsheet multiple times to continue working on the assignment.

b. The Google spreadsheet interface is rather buggy and often faieds to load scraped data. If there were a more reliable spreadsheet program that could read XPath, it would be great to use that!

It might also be useful to investigate other possible uses for XPath in the assignment, because at the moment I am not exactly sure for what else XPath can be used. I also noticed that XPath can be incorporated into programming and Java Script. It would be great if future students could investigate that.

c. Yes, I think that anyone should be able to legally gather and use data on a free to use, publicly accessible site, like Craigslist. On a site like Craigslist, there is no expectation of privacy on behalf those people posting an advertisement, as least with regards to the information they are choosing to publically share. I do not think that Craigslist itself has any more proprietary claim over the information uploaded to the site than the users themselves because the service and data they provide access to is public and requires no login or active agreement to terms of service to view. I do see an issue, however, if people are using the information they harvest from a site like Craigslist for commercial purposes. That seems to me like some sort of proprietary assertion over the data that they have gleaned for free, which seems shady and inequitable.

Rachel

a. The most difficult part of the assignment was manipulating large amounts of data in Google spreadsheets. Sometimes the xpath commands in the spreadsheet would have difficulty loading large amount of scraped data. In addition, Craigslist makes it difficult to scrape certain information from webpage tags. For example, addresses are not always embedded to the same place in code for each web page. Information about addresses were either nested in the page titles tag or within the ‘mapaddress’ tag. As a result, standardizing address information in Google spreadsheets introduced a bit of a challenge.

b. It would be good to learn how to incorporate Javascript into the assignment’s goals. For the purposes of this assignment we used Python because it was able to read the .CSV more easily that Javascript. However, it would be great to apply what we have learned in JavaScript to the goals of this assignment.

c. I believe that it should be legal to scrape data from websites such as Craigslist. Craigslist is free to use and comprises of data from and for the public. Anyone should be able to gather and use the data because it is voluntarily shared and accessed for free. Scraping housing listings from Craigslist provides the data for new projects that can better improve and facilitate an individual’s search for desired apartment rentals.