Technical Module Design Brief

Chris Henrick
Thesis Studio One
Scott Pobiner, Louisa Campbell
4/19/2014

Design Questions

- How can technology bridge the communication gap around the issues of affordable housing and displacement between stakeholders that speak separate languages such as elected city officials, grassroots organizers and civic technologists?
- What are the unmet needs of community groups that are working towards promoting truly affordable housing and the prevention of displacement of working class and low-income residents in New York City?

Research

Having met with two local community groups in Brooklyn; The Northwest Bushwick Community Group (NBCG) and Movement to Protect the People (MTOPP), needs and communication gaps from both groups were identified.

The Northwest Bushwick Community Group

The NBCG was established several years ago after the council woman serving the Community Board that encompasses the Bushwick area informed residents of planned up-zoning and a luxury condo development. Following a hearing on this matter residents of Bushwick, Brooklyn formed the NBCG to advocate for the retention of truly affordable housing and prevention of the displacement of residents in Bushwick. Following the group's organization NBCG met with

students from the Parsons Urban Ecologies program for a workshop on development and displacement. This led to further organizing efforts including the participatory mapping of vacant land, vacant buildings, new developments and developments in progress within the Bushwick area, which the author took part in.

Following this workshop the group worked with a GIS analyst to establish an online community map of Bushwick that displays data relating to housing and development in the Bushwick area. Currently the map has limited functionality and lacks clarity in what it attempts to communicate to an un-specified audience. A need exists for this map to be re-designed to convey the organization's narrative relating to development, gentrification and development in Bushwick to a specific audience.

Movement to Protect the People

MTOPP was born earlier in 2014 after a local resident began organizing to prevent the up-zoning of a commercial corridor along Empire Boulevard in the neighborhood of Prospect Lefferts Gardens located just south of Crown Heights and east of Prospect Park. After meeting with MTOPP the organization requested the transaction histories of property along Empire Blvd to identify potential new development trends. The author then stated he would assist MTOPP with acquiring the transaction data.

Project Concept

Currently the author is working with both community groups on an on going basis. For the purpose of this design brief only the work for MTOPP is being addressed as the work for NBCG has not lead to a prototype as of yet.

The data requested by MTOPP is located online via a city website called ACRIS. The method by which a user requests and acquires property transaction records through ACRIS is convoluted and the data is not accessible in a format which can be opened with spreadsheet software such as Microsoft Excel. The first part of this module's prototype was to identify block and lot values for the 34 properties in the study area along Empire Boulevard and then write a python script in order to

scrape their transaction history records. Each of these transaction history records was then outputted to a CSV file and submitted to MTOPP.

Following the successful web scraping the author conceptualized a web-app to make obtaining property transaction histories in bulk easier for a user. The goal of such a tool would be to improve access to this information to community groups. Thus such groups could empower themselves by being able to look up and download all property transaction histories for their neighborhood. A functional web-app has been started but has not been finished due to time restrictions and programming obstacles.

Methodology

The steps for web-scraping were as follows:

1. Meet with MTOPP to identify the organization's needs.
2. Use NYC's MapPluto data with a GIS to determine the Borough, Block and Lot numbers for the requested properties along Empire Blvd between South Crown Heights and Prospect Lefferets Garden in Brooklyn, NY.
3. Output the desired data from the GIS to a CSV file which can then be read by a python script to automate the reading and extraction of Borough, Block and Lot numbers.
4. Using Python, mimic a web browser and submit POST requests to the ACRIS server for each property in the CSV file. Randomize a period of time for the script to "sleep" between each POST request to mimic a typical website user and to not overload the ACRIS servers.
5. Output the transaction records for each property to a CSV file.
6. Send the files to MTOPP.

The steps for the implementation of a web app were as follows:

1. Create a UI that allows a user to draw a polygon around a desired area on an interactive map. This area will be used to perform an intersection on tax-lots that provide Borough, Block and Lot numbers for retrieving building transaction history from ACRIS data.

2. Import the transaction history data from ACRIS via NYC Open Data into a Postgres database on a web server that can be queried via a PostGIS.
3. Once the data is queried output it to a CSV format that can be downloaded by the user.

Findings and Next Steps

Due to restraints in the author's knowledge of server side programming and the allotted time in which this brief was to be prepared the web app was not finished. However the next steps are identified as follows:

With help from BetaNYC determine how to import the ACRIS bulk data from the NYC Open Data portal into a Postgres database on a web server. Then finish coding the web app so that the data can be queried by a user through a map interface and returned as downloadable CSV files.

Appendix

Part 1: Web Scraping with Python

Sample HTML table obtained from ACRIS through a POST request:

New York City Department of Finance
Office of the City Register

HELP
[Click help for additional instructions]
Selecting a help option will open new window

Current Search Criteria:

Borough: BROOKLYN / KINGS
Block: 1306
Lot: 35 **Unit:** N/A
Date Range:
Document Class: All Document Classes

Search Results By Parcel Identifier

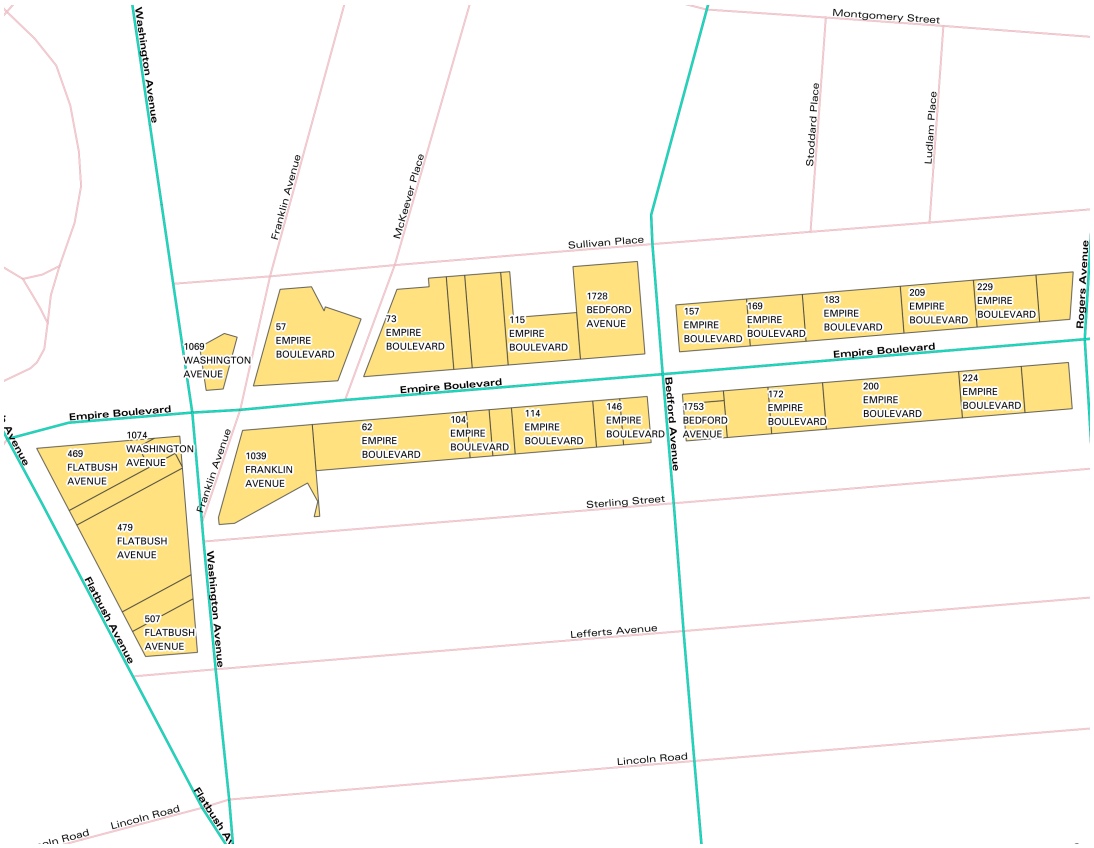
Records 1 - 7 << previous next >> Max Rows 10 [Search Options] [New BBL Search] [Edit Current Search] [View Tax Map] [Print Index]

View	Reel/Pg/File	CRFN	Lot	Partial	Doc Date	Recorded / Filed	Document Type	Pages	Party1	Party2	Party 3/ Other	More Party 1/2 Names	Corrected/ Remarks	Doc Amount
DET IMG		2014000026943	35	PARTIAL LOT	1/21/2014	1/23/2014 10:41:57 AM	INITIAL UCC1	5	WORLD LAUNDRY REALTY, INC.	FINANCIAL PACIFIC LEASING, INC.				0
DET IMG	5199/1784		35	ENTIRE LOT		6/27/2001	SUNDRY AGREEMENT	4	MDL EQUIPMENT CORP	EMPIRE ESTATES LLC				0
DET IMG	4208/626		35	ENTIRE LOT		5/29/1998	LEASE	3	MDL EQUIPMENT CORP	EMPIRE ESTATES LLC				0
DET IMG	2426/1997		35	ENTIRE LOT		7/31/1989	SATISFACTION OF MORTGAGE	2	MDL EQUIPMENT CORP	MOSLIN, SEYMOUR		✓		0
DET IMG	1068/1216		35	ENTIRE LOT	4/26/1979	4/26/1979	MORTGAGE	4	MDL EQUIPMENT CORP	MOSLIN SEYMOUR		✓		0
DET IMG	1068/1210		35	ENTIRE LOT	4/26/1979	4/26/1979	DEED	2	MOSLIN SEYMOUR	MDL EQUIPMENT CORP		✓		0
DET IMG	931/716		35	ENTIRE LOT	7/12/1977	7/12/1977	DEED	2	73 REALTY CORP	MOSLIN SEYMOUR		✓		0

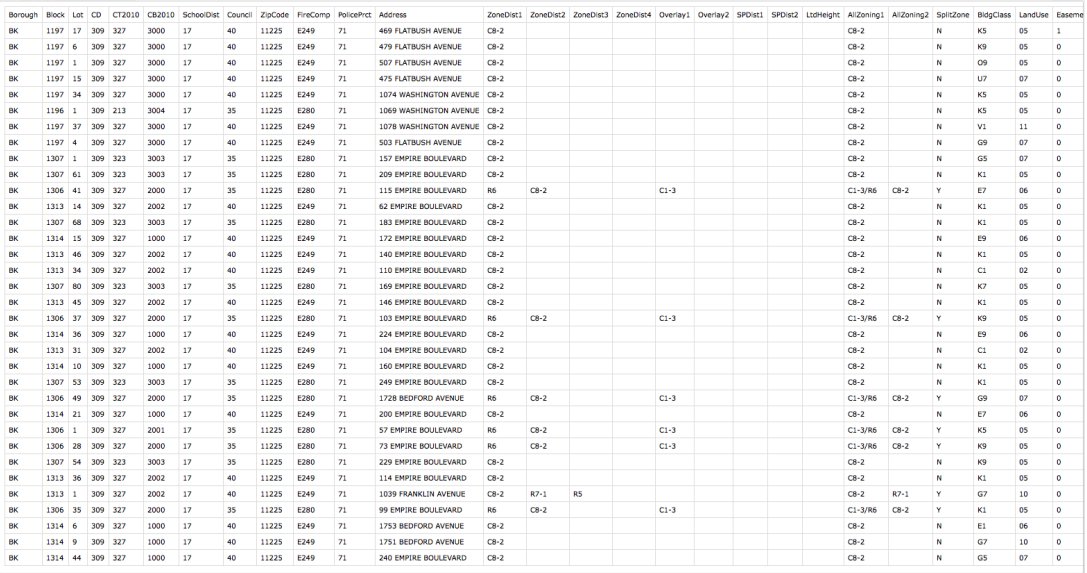
Search Options | New Parcel Identifier Search | Edit Current Search | View Tax Map

Go To: Finance Home Page | NYC.gov Home Page | Contact NYC.gov | FAQs | Privacy Statement | Site Map

Screen shot from QGIS showing study area in Prospect Lefferts Gardens:



Sample of data outputted from Map-PLUTO using QGIS:



Borough	Block	Lot	CD	CT2010	CB2010	SchoolDist	Council	ZipCode	FireComp	PolicePrct	Address	ZoneDist1	ZoneDist2	ZoneDist3	ZoneDist4	Overlay1	Overlay2	SPDist1	SPDist2	LtdHeight	AllZoning1	AllZoning2	SplitZone	BldgClass	LandUse	Easeme
BK	1197	17	309	327	3000	17	40	11225	E249	71	469 FLATBUSH AVENUE	C8-2									C8-2		N	K5	05	1
BK	1197	6	309	327	3000	17	40	11225	E249	71	479 FLATBUSH AVENUE	C8-2									C8-2		N	K9	05	0
BK	1197	1	309	327	3000	17	40	11225	E249	71	507 FLATBUSH AVENUE	C8-2									C8-2		N	O9	05	0
BK	1197	15	309	327	3000	17	40	11225	E249	71	475 FLATBUSH AVENUE	C8-2									C8-2		N	U7	07	0
BK	1197	34	309	327	3000	17	40	11225	E249	71	1074 WASHINGTON AVENUE	C8-2									C8-2		N	K5	05	0
BK	1196	1	309	213	3004	17	35	11225	E280	71	1069 WASHINGTON AVENUE	C8-2									C8-2		N	K5	05	0
BK	1197	37	309	327	3000	17	40	11225	E249	71	1078 WASHINGTON AVENUE	C8-2									C8-2		N	V1	11	0
BK	1197	4	309	327	3000	17	40	11225	E249	71	503 FLATBUSH AVENUE	C8-2									C8-2		N	G9	07	0
BK	1307	1	309	323	3003	17	35	11225	E280	71	157 EMPIRE BOULEVARD	C8-2									C8-2		N	G5	07	0
BK	1307	61	309	323	3003	17	35	11225	E280	71	209 EMPIRE BOULEVARD	C8-2									C8-2		N	K1	05	0
BK	1306	41	309	327	2000	17	35	11225	E280	71	115 EMPIRE BOULEVARD	R6	C8-2			C1-3					C1-3/R6	C8-2	Y	E7	06	0
BK	1313	14	309	327	2002	17	40	11225	E249	71	62 EMPIRE BOULEVARD	C8-2									C8-2		N	K1	05	0
BK	1307	68	309	323	3003	17	35	11225	E280	71	183 EMPIRE BOULEVARD	C8-2									C8-2		N	K1	05	0
BK	1314	15	309	327	1000	17	40	11225	E249	71	172 EMPIRE BOULEVARD	C8-2									C8-2		N	E9	06	0
BK	1313	46	309	327	2002	17	40	11225	E249	71	140 EMPIRE BOULEVARD	C8-2									C8-2		N	K1	05	0
BK	1313	34	309	327	2002	17	40	11225	E249	71	110 EMPIRE BOULEVARD	C8-2									C8-2		N	C1	02	0
BK	1307	80	309	323	3003	17	35	11225	E280	71	169 EMPIRE BOULEVARD	C8-2									C8-2		N	K7	05	0
BK	1313	45	309	327	2002	17	40	11225	E249	71	146 EMPIRE BOULEVARD	C8-2									C8-2		N	K1	05	0
BK	1306	37	309	327	2000	17	35	11225	E280	71	103 EMPIRE BOULEVARD	R6	C8-2			C1-3					C1-3/R6	C8-2	Y	K9	05	0
BK	1314	36	309	327	1000	17	40	11225	E249	71	224 EMPIRE BOULEVARD	C8-2									C8-2		N	E9	06	0
BK	1313	31	309	327	2002	17	40	11225	E249	71	104 EMPIRE BOULEVARD	C8-2									C8-2		N	C1	02	0
BK	1314	10	309	327	1000	17	40	11225	E249	71	160 EMPIRE BOULEVARD	C8-2									C8-2		N	K1	05	0
BK	1307	53	309	323	3003	17	35	11225	E280	71	249 EMPIRE BOULEVARD	C8-2									C8-2		N	K1	05	0
BK	1306	49	309	327	2000	17	35	11225	E280	71	1728 BEDFORD AVENUE	R6	C8-2			C1-3					C1-3/R6	C8-2	Y	G9	07	0
BK	1314	21	309	327	1000	17	40	11225	E249	71	200 EMPIRE BOULEVARD	C8-2									C8-2		N	E7	06	0
BK	1306	1	309	327	2001	17	35	11225	E280	71	57 EMPIRE BOULEVARD	R6	C8-2			C1-3					C1-3/R6	C8-2	Y	K5	05	0
BK	1306	28	309	327	2000	17	35	11225	E280	71	73 EMPIRE BOULEVARD	R6	C8-2			C1-3					C1-3/R6	C8-2	Y	K9	05	0
BK	1307	54	309	323	3003	17	35	11225	E280	71	229 EMPIRE BOULEVARD	C8-2									C8-2		N	K9	05	0
BK	1313	36	309	327	2002	17	40	11225	E249	71	114 EMPIRE BOULEVARD	C8-2									C8-2		N	K1	05	0
BK	1313	1	309	327	2002	17	40	11225	E249	71	1039 FRANKLIN AVENUE	C8-2	R7-1	R5							C8-2	R7-1	Y	G7	10	0
BK	1306	35	309	327	2000	17	35	11225	E280	71	99 EMPIRE BOULEVARD	R6	C8-2			C1-3					C1-3/R6	C8-2	Y	K1	05	0
BK	1314	6	309	327	1000	17	40	11225	E249	71	1753 BEDFORD AVENUE	C8-2									C8-2		N	E1	06	0
BK	1314	9	309	327	1000	17	40	11225	E249	71	1751 BEDFORD AVENUE	C8-2									C8-2		N	G7	10	0
BK	1314	44	309	327	1000	17	40	11225	E249	71	240 EMPIRE BOULEVARD	C8-2									C8-2		N	G5	07	0

Final Python code used for ACRIS web scraping:

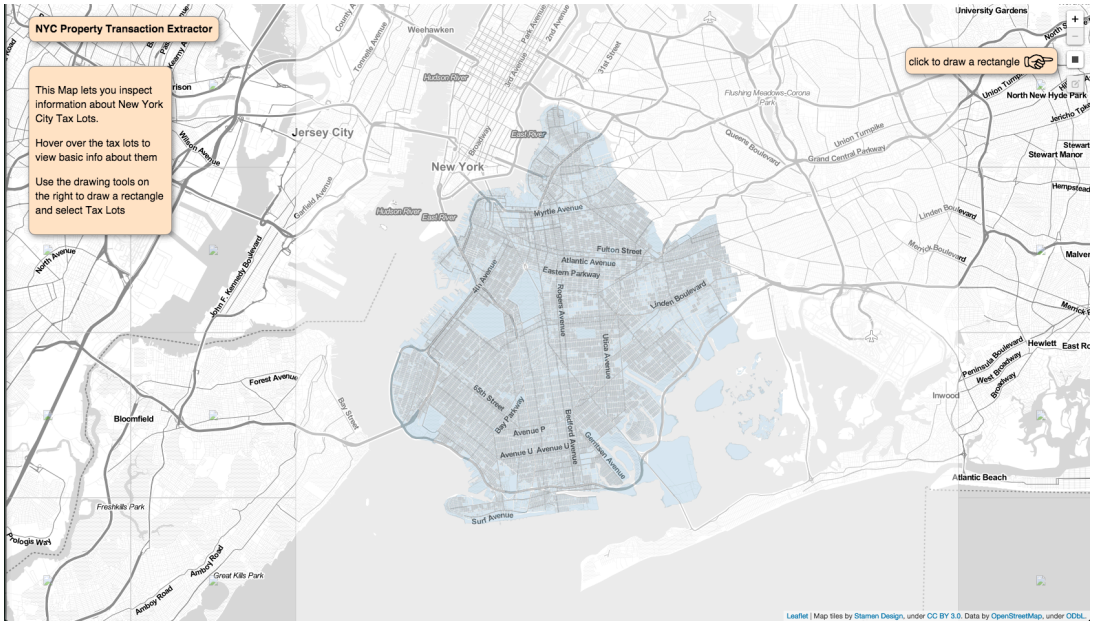
```
from sys import argv
import csv
import requests
import bs4
import time
import random

script, filename = argv

# f = open(filename)

url = "http://a836-acris.nyc.gov/DS/DocumentSearch/BBLResult"
headers = {'User-Agent' : 'Mozilla/5.0'}

count = 0

with open(filename, 'rb') as f:
    reader = csv.reader(f)
    next(reader, None)
```

```
    try:
        for row in reader:
            print row
            # example url from arcis http://a836-acris.nyc.gov
/bblsearch/bblsearch.asp?borough=3&block=1306&lot=35
            block = row[1]
            lot = row[2]
            address = row[11]
            #print  "the block is %s and the lot is %s" % (blo
ck, lot)
            url2post = "http://a836-acris.nyc.gov/bblsearch/bb
lsearch.asp?borough=3&block=%s&lot=%s" % (block,lot)
            print "the address is %s and the url is: %s" % (ad
dress, url2post)

            data = {
            'hid_borough':'3',
            'hid_borough_name':'BROOKLYN / KINGS',
            'hid_block':block,
            'hid_block_value': block,
            'hid_lot':lot,
            'hid_lot_value': lot,
            'hid_doctype_name':'All Document Classes',
            'hid_max_rows':'10',
            'hid_page':'1',
            'hid_SearchType':'BBL',
            'hid_ISIntranet':'N'
            }
            print data

            t = open(address + ".csv", 'w+')
            # write column headers
            t.write("Reel/Pg/File,CRFN, Lot, Partial, Doc Date
, Recorded / Filed, Document Type, Pages, Party1, Party2, Part
y3 / Other, More Party 1/2 Names, Corrected / Remarks, Doc Amo
unt\n")

            response = requests.post(url, headers=headers,data
=data)
            soup = bs4.BeautifulSoup(response.text)
```

```
            print response

            table = soup.find(attrs={"cellspacing":"1","width"
:"100%"})

            # iterate over table
            for row in table.find_all('tr')[1:]:

                for col in row.find_all('td')[1:]:
                    # f = col.find_all('font')

                    for f in col.find_all('font'):
                        value = f.string
                        print value

                        try:
                            # print value.strip()
                            value = value.replace(',','')
                            t.write(value.strip())
                            t.write(',')

                        except Exception:
                            t.write('*,')
                            pass
                        count +=1
                    if count !=0 and count % 14 == 0:
                        t.write('\n')

            t.close()
            time.sleep(random.randrange(32,48))
    except csv.Error as e:
        sys.exit('file %s, line %d: %s' % (filename, reader.li
ne_num, e))
```

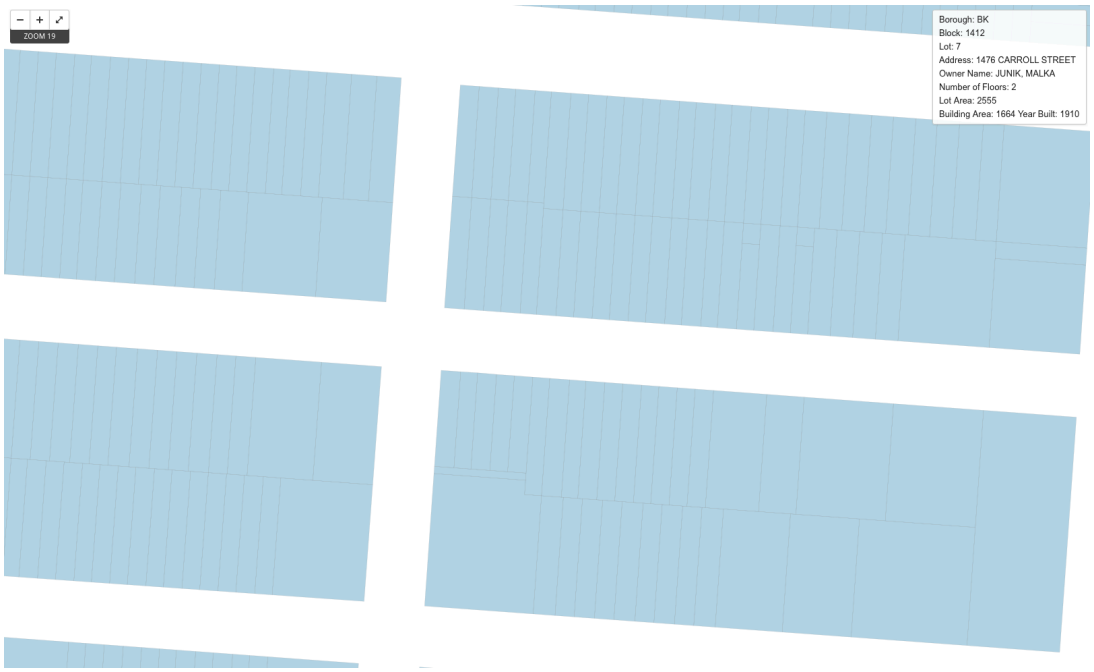
Part II: Designing a Web App to Automate ACRIS data retrieval:

Sample web map UI:

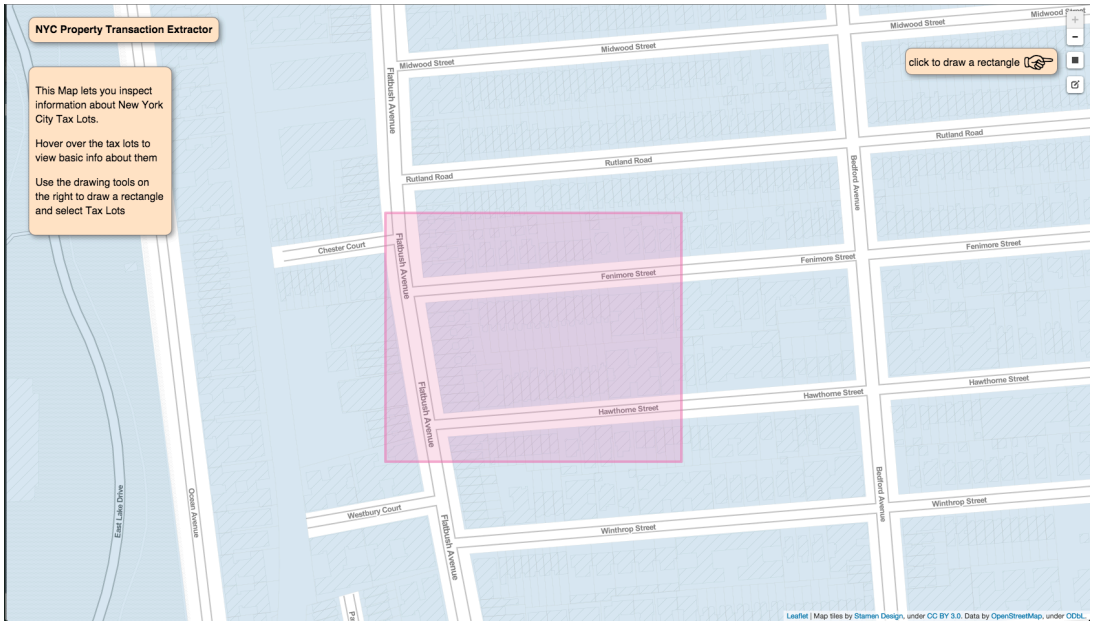


Zooming in to inspect Block and Lots

Hovering the mouse over a tax-lot reveals it's Borough, Block, Lot, Address, Owner Name, Number of Floors, Lot Area, Building Area and Year Built.



Using the Leaflet Draw plug-in to create a polygon that will select the desired tax-lots.



Code for app's front-end:

Index.html

```
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome
=1">
    <meta name="description" content="">
    <meta name="author" content="">
    <title></title>
    <link rel="stylesheet" type="text/css" href="css/leaflet.c
ss">
    <link rel="stylesheet" type="text/css" href="css/styles.cs
s">
    <link rel="stylesheet" type="text/css" href="css/leaflet.d
raw.css">
  </head>
  <body>
    <header class="box">NYC Property Transaction Extractor</he
ader>
    <div id="map"></div>
    <div id="about" class="box">
      <p>This Map lets you inspect information about New York
City Tax Lots.</p>
      <p>Hover over the tax lots to view basic info about them
</p>
      <p>Use the drawing tools on the right to draw a rectangl
e and select Tax Lots</p>
    </div>
    <div class="box info1">
      <p>click to draw a rectangle <span>&#9758;</span></p>
    </div>

    <script src="js/jquery-2.1.1.min.js"></script>
    <script src="js/underscore-min.js"></script>
    <script src="js/leaflet.js"></script>
    <script src="js/leaflet.draw.js"></script>
    <script src="js/leaflet.utfgrid.js"></script>
    <script type="text/javascript" src="http://maps.stamen.com
/js/tile.stamen.js?v1.2.3"></script>
    <script src="js/main.js"></script>
```

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

main.js

```
var app = app || {};

app.map = ( function (w, d) {

  var e = {
    map : null,
    lots : null,
    test : null,
    utfGrid : null
  };

  // render the map
  var initMap = function(){
      console.log('initMap called');
      var config = {
          baselayer : new L.StamenTileLayer("toner-lite"),
          initLatLng : new L.LatLng(40.7, -74),
          zoom : 12,
          minZoom : 12,
          maxZoom : 18,
          zoomControl : false,
          attributionControl : true,
          maxBounds : L.latLngBounds([40.539373,-74.117203],[4
0.771182,-73.798599])
      };

      e.map = L.map('map', config);
      e.map.addLayer(config.baselayer);
      e.map.setView(config.initLatLng, config.zoom);

      e.test = L.tileLayer('http://localhost:8888/v2/nyc_pluto
_test/{z}/{x}/{y}.png', {opacity: 0.5});
      e.map.addLayer(e.test);
```

```
      e.utfGrid = new L.UtfGrid('http://localhost:8888/v2/nyc_
pluto_test/{z}/{x}/{y}.grid.json?callback={cb}', {
        resolution: 4
      });

      e.utfGrid.on('mouseover', function(e){ info.update(e);})
.on('mouseout', function(e){ info.update();})

      var info = L.control();
      info.options.position = 'bottomright';
      info.onAdd = function (map) {
          this._div = L.DomUtil.create('div', 'info'); // crea
te a div with a class "info"
          this.update();
          return this._div;
      };

      info.update = function (props) {
        this._div.innerHTML = "<h4>Block Lot Info</h4>" +  (pr
ops ?
        "<values><b>" + props.data.Block + "</b><br>Lot <rank>
" + props.data.Lot+"</rank></values>"
        : 'Hover over a tax lot');
      };

      // e.map.addLayer(e.utfGrid)
      //           .addControl(info);


      new L.control.zoom({position: 'topright'}).addTo(e.map);

      var editableLayers = new L.FeatureGroup();

      var options = {
          position: 'topright',
          draw: {
              polyline: false,
              polygon: false,
              circle: false, // Turns off this drawing tool
              rectangle: {
```

```
                shapeOptions: {
                    clickable: false
                }
            },
            marker: false
        },
        edit: {
            featureGroup: editableLayers, //REQUIRED!!
            remove: false
        }
    };

    var drawControl = new L.Control.Draw(options);

    e.map.addLayer(editableLayers);
    e.map.addControl(drawControl);
    e.map.on('draw:created', function (e) {
        var type = e.layerType,
            layer = e.layer;

        if (type === 'marker') {
            layer.bindPopup('A popup!');
        }

        editableLayers.addLayer(layer);
    });

};

// vector data overlay
var fetchData = function() {
  // pop-up content
  var pc = function(feature) {
    return;
  };

  var style = {
    color: '#03f',
    weight: 3,
    opacity: 0.5,
```

```
      fill: '#fff',
      fillOpacity: '0.2'
    };

    $.getJSON('http://localhost:3000/amenities', function(d) {
        console.log('the data is: ', d);
        e.lots = L.geoJson(d, {
          style : style
        }).addTo(e.map);

    });
  };

  var init = function() {
    console.log('app.map init called');
    initMap();
    //fetchData();
  };

  return {
    init : init,
    elements : e
  };

})( window, document );

window.addEventListener('DOMContentLoaded', app.map.init);
```