

# Mapping Out Crime in Washington D.C.

Ian Helm

George Mason University  
Dept of Geography and Geosciences  
4400 University Drive, MS 6C3  
Fairfax, VA 22030-4444  
ihelm@gmu.edu

## ABSTRACT

In this paper, I will be presenting a web mapping application meant for selecting and displaying crime data within Washington D.C. in 2017. The crime data itself was stored and imported from CartoDB. The web mapping application was created in Codepen and using Leaflet. This process allowed the development of comprehensive layer functionalities such as marker clusters, basemap selection, click functionality, and dropdown boxes. The result would be an interactive map of crime incidents within Washington D.C. When the program is initially loaded, all crimes committed in the city would be displayed and grouped with a marker cluster. Users can then interact with the dropdown box to select through specific types of offenses. Users can also zoom down to a more local level and click on each individual point. Clicking on these points ends up showing the date and time that each specific incident was committed.

## Categories and Subject Descriptors

D.3.3 [Programming Languages]: PostGIS, PostgreSQL, Javascript

## General Terms

Algorithms, Measurement, Documentation, Security, Human Factors, Languages

## Keywords

Crime, PostGIS, Dropdown, Javascript, Web mapping, Leaflet, Click Functionality

## 1. INTRODUCTION

Crime is a prevalent problem within Washington D.C's eight wards. Washington D.C. was recently ranked as the 16<sup>th</sup> most dangerous city within the United States [1]. Reports on such incidents have been collected and available to the public on the internet.

The objective of this project is to use open source libraries and applications to build a web mapping application. This application would allow a user to either view the total number of crimes committed in Washington D.C. or narrow it down to specific offenses. Furthermore, this project allows a user to click on each recorded point and view the month, day, and time that each incident took place on.

## 2. DC CRIME DATA

Reported crime incidents within the Washington D.C. Metropolitan Area are recorded and made publicly available for free on websites such as Open Data DC [2]. This website allows users to publicly

view and download shapefiles, tables, and other forms of Geographic Information Systems (GIS) Data.

For this project, I downloaded the Crime Incidents in 2017 dataset from Open Data DC [3], which consists of all recorded crime incidents in Washington D.C. between January 1 and December 31 of 2017. The data can be download in various formats, including the shapefile format that I have chosen for this assignment. I chose the data from 2017 since 2018 is still in progress and thus does not provide a complete report of crime incidents within Washington D.C.

When loaded, each row of data represents a specific crime incident that took place between January 1, 2017 and December 31, 2017. These rows give out vital information such as:

- Report\_dat – The year, month, day, and time that the incident occurred
- Shift – The shift that the crime incident occurred during
- Method – The method that was used to commit the reported crime (knife, gun, others)
- Ward – The ward that the incident occurred in
- Offense – The type of crime that was committed

Offenses for crimes are classified to the following distinct categories within the dataset:

- Motor Vehicle Theft
- Assault W/Dangerous Weapon
- Arson
- Theft F/Auto
- Homicide
- Sex Abuse
- Burglary
- Theft
- Robbery

This means that we have several related yet distinct categories that users can select and filter through when viewing crime incidents within Washington D.C. throughout 2017.

## 3. ARCHITECTURE

After analyzing the Crime Incidents in 2017 dataset itself, I will be explaining how this data was loaded, categorized, and displayed onto a basic web mapping application. The data itself was first loaded on CartoDB. After an SQL query was implemented, it

would be loaded onto Codepen. Javascript, HTML, and CSS were used for creating an interactive web map using that data.

### 3.1 CARTO

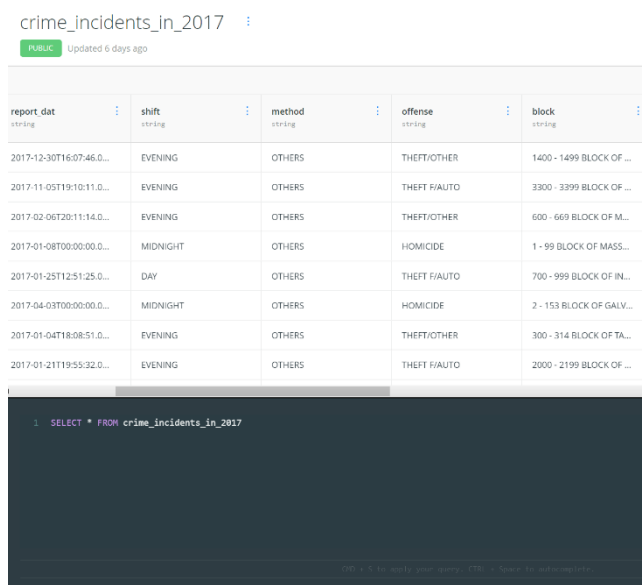
The crime data itself was downloaded as an unzipped shapefile, and then loaded onto CartoDB [4]. Once that was completed, I used PostGIS to select the data set based on any category that I wanted. However, for this project all I needed to do in CartoDB was to implement the following PostGIS query:

```
SELECT * FROM crime_incidents_in_2017
```

This is the default query that is provided within any dataset that is initially loaded onto CartoDB. By implementing this query, all the data is selected without being narrowed down to any specific categories.

This PostGIS query is all I need to import into Codepen later. With all the crime incidents and their related offenses selected, they can be separated into distinct categories after the SQL query is imported into Codepen. This will allow layer functionality to be implemented on the final web map application and thus allow the user to select specific offenses to be displayed on the map of Washington D.C.

Figure 1. Shows the result of the CartoDB data query prior to implementing.



The screenshot shows the CartoDB interface for a dataset named 'crime\_incidents\_in\_2017'. It displays a table with the following columns: report\_dat, shift, method, offense, and block. Below the table, a SQL query is shown in a dark-themed editor: 'SELECT \* FROM crime\_incidents\_in\_2017'.

report_dat	shift	method	offense	block
2017-12-30T16:07:46.0...	EVENING	OTHERS	THEFT/OTHER	1400 - 1499 BLOCK OF ...
2017-11-05T19:10:11.0...	EVENING	OTHERS	THEFT FIAUTO	3300 - 3399 BLOCK OF ...
2017-02-06T20:11:14.0...	EVENING	OTHERS	THEFT/OTHER	600 - 669 BLOCK OF M...
2017-01-08T00:00:00.0...	MIDNIGHT	OTHERS	HOMICIDE	1 - 99 BLOCK OF MASS...
2017-01-25T12:51:25.0...	DAY	OTHERS	THEFT FIAUTO	790 - 999 BLOCK OF IN...
2017-04-03T00:00:00.0...	MIDNIGHT	OTHERS	HOMICIDE	2 - 153 BLOCK OF GALV...
2017-01-04T18:08:51.0...	EVENING	OTHERS	THEFT/OTHER	300 - 314 BLOCK OF TA...
2017-01-21T19:55:32.0...	EVENING	OTHERS	THEFT FIAUTO	2000 - 2199 BLOCK OF ...

**Figure 1. The CartoDB dataset with its associated PostGIS query**

### 3.2 USING CODEPEN

The next step was to load up CodePen [5], which allows me to create web applications using scripting and programming

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Conference '10, Month 1–2, 2010, City, State, Country.

Copyright 2018 ACM 1-58113-000-0/00/0010 ...\$15.00.

languages such as HTML, CSS, and JavaScript. In addition to general JavaScript, this project required the widespread use of Leaflet. Leaflet is an open-source JavaScript library that provides the necessary code structure to create interactive web mapping applications [6]. The very first step in using leaflet was to set a variable “map” which would be centered on the coordinates [38.9072, -77.0369] and set to a zoom level of 11. This will allow the web mapping application by default to be loaded on the coordinate of Washington D.C. and be zoomed out to the point where the entire city is visible to the user.

After the map itself was created and set to the proper default coordinates, I added three Leaflet-provider basemaps to the program, “OSM B&W”, “OpenStreetMap\_Mapnik” and “OpenStreetMaps\_DE” [7]. Furthermore, these basemaps would then be grouped together so that the user would be able to choose between them.

#### 3.2.1 Marker Clusters

Following the creation of the map itself, I created marker clusters that would allow points to be automatically grouped based on how much I zoom in or out.

After the creation of the MarkerClusterGroup, I set various parameters for the geojsonMarkerOptions. This ranged from the radius, the color of the fill and border of each point, and the opacity of each colored point. Since I only needed one color, I selected one by using the HTML color picker at w3schools.com [8]. This allowed me to experiment with various color combinations until I picked one that I was satisfied with. This would stylize each point on the web application as opposed to simply using the default marker object style.

#### 3.2.2 Queries and Popups

Once the map and marker clusters were created, the actual data from CartoDB needed to be implemented into leaflet. This was done by creating the following code lines:

```
var querysystem =
```

```
"https://ianhelm.carto.com/api/v2/sql?format=GeoJSON&q=SELECT * FROM crime_incidents_in_2017";
```

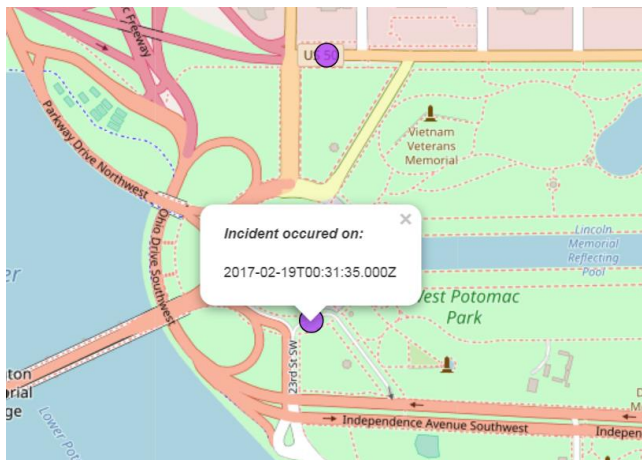
```
var query = querysystem;
```

As mentioned earlier, this program simply called an SQL API key for the unfiltered crime data that was originally uploaded to CartoDB [9]. The implementation of layer functionality later would allow me to select and filter through the crime data based on certain attributes.

With the call of the SQL API key into Leaflet, I felt that it was necessary to also implement click functionality into my web application. This was done through writing the following code line:

```
layer.bindPopup("<h4><i> Incident occurred on:</i></h4> " +  
feature.properties.report_dat + "</p>");},
```

As demonstrated, the layer.bindPopup variable uses the report\_dat attribute from the 2017 DC crime dataset. This means that clicking on each point will end up displaying the year, month, date, and time that the crime occurred. Figure 2 below demonstrates this, as clicking on the screenshotted point tells me that the incident occurred on October 11, 2017 at 17:09:29



**Figure 2. The date displayed from clicking on a crime incident point**

### 3.2.3 Implementing Dropboxes

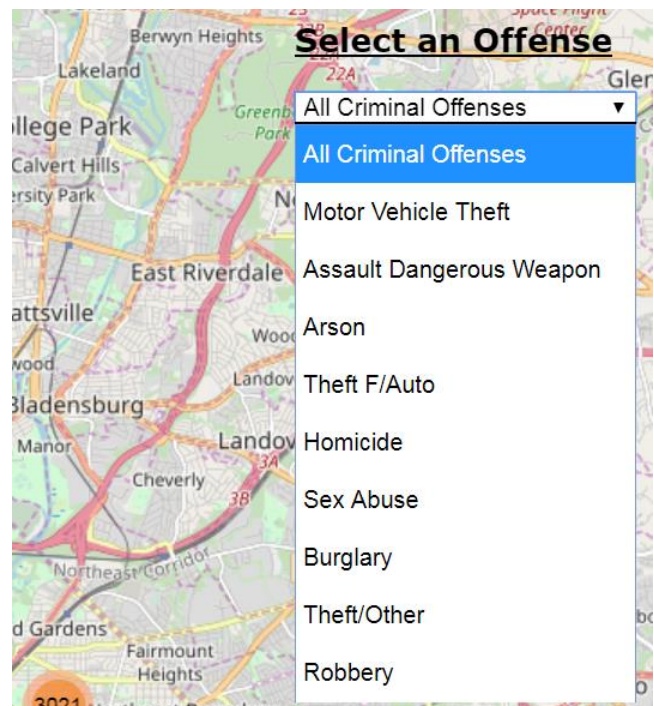
As mentioned earlier in the paper, all the crime data from CartoDB was selected and then imported into Codepen. Later, it would be organized through the use of layer functionality. To be specific, I implemented dropdown boxes in my web mapping application.

Crime incidents would be categorized in the dropdown box by the type of offense that was committed (Motor Vehicle Theft, Arson, etc.). This would allow the user to either view all points representing all crimes committed by default, or to narrow it down to specific types of offenses.

First, a default if statement was declared in the program. This would allow all crime incidents from 2017 to be displayed on the map by default when the web application is initially loaded. Once that was completed, 9 additional “else-if” statements had to be added. This would effectively mean the creation of individual layers to be added to the dropdown box. The following code below represents some of the 9 else-if statements creating a layer and associated SQL query:

```
else if (e.target.value == "layer1") {
    query = quystem + " WHERE offense = 'MOTOR VEHICLE THEFT'";
} else if (e.target.value == "layer2") {
    query = quystem + " WHERE offense = 'ASSAULT W/DANGEROUS WEAPON'";
} else if (e.target.value == "layer3") {
    query = quystem + " WHERE offense = 'ARSON'";
```

In addition to the else-if statements, the dropdown box itself was added to the application. It would be positioned on the top-right corner of the map, with a title above that states, “Select an Offense”. There is a total of 10 options in the box, and each layer option would have a label associated with it. This would give the user a clear description of what types of crime incidents they would be selecting. Figure 3 demonstrates the finished dropdown box that would implement layer functionality and grant the ability to select through crime data.

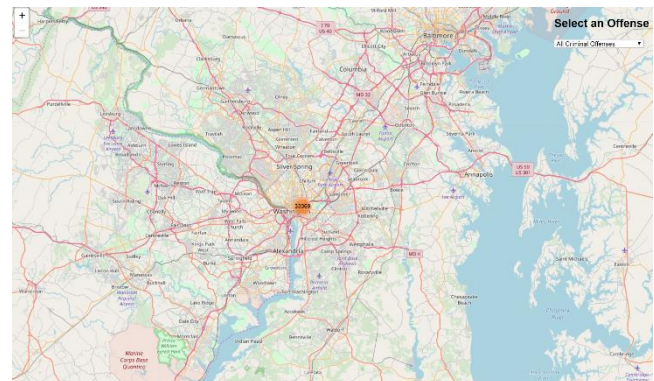


**Figure 3. The dropdown box with available types of crimes**

## 4. ANALYZING CRIME IN D.C.

Following the creation of the web map, it can now be used to analyze the number and location of crimes committed within Washington D.C. in 2017. Crimes can also be narrowed down to specific types of offenses. Furthermore, the date that the crime occurred can be displayed by clicking on the point

Figure 4 displays the initial map, which is zoomed out to show the total number of crimes that occurred in D.C.



**Figure 4. The initial result from loading up the web mapping application**

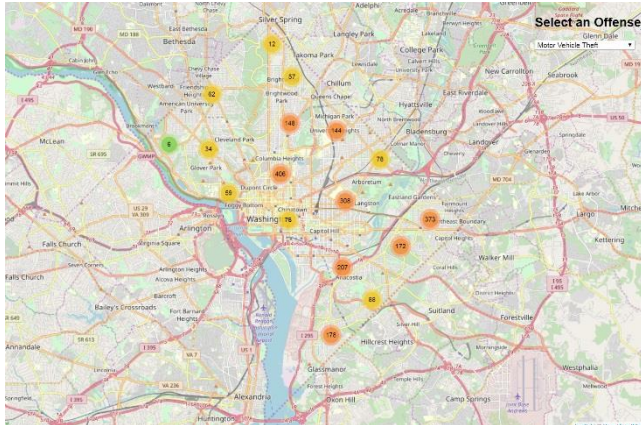
The implementation of marker clusters lets the map automatically group similar values and display the number of values within that group. This allows us to see the number of crimes that were committed based on the extent of the zoom. Figure 4 demonstrates that 33069 total crimes were committed in Washington D.C. throughout 2017.

Based on the positioning of option layers within the dropdown box, the map will always be set by default to the “All Criminal Offenses” option. This means that all general crimes in D.C. will be displayed



upon initially loading the web mapping application. However, the dropdown box has 9 additional options available for specific crime offenses.

For example, selecting the “Motor Vehicle Theft” option will clearly display all 2017 crime incidents where the offense committed was motor vehicle theft. The implementation of marker clusters also helps display the number of incidents for each specific offense. When zooming out to the citywide level, the marker clusters display that 2408 motor vehicle theft incidents occurred in Washington D.C. throughout 2017. As one zooms in a closer level, the marker clusters will stay grouped, but automatically break off into different regions (e.g. Georgetown, National Mall). Figure 5 demonstrates the result from selecting Motor Vehicle Theft on the dropdown box.



**Figure 5. The marker clusters of Motor Vehicle Theft incidents in D.C.**

In addition to this, we can also zoom down to individual points and understand the proximity of motor vehicle theft incidents in a localized region. There are still a few marker cluster groups that exist at a local zoom. This could indicate that a block within Washington D.C. has seen multiple motor vehicle theft incidents throughout the year. Figure 6. Demonstrates the results from selecting Motor Vehicle Theft on the dropdown box and then zooming down to a local level.



**Figure 5. The local points/clusters of Motor Vehicle Theft incidents around Capitol Hill**

## 5. FUTURE APPLICATIONS

As demonstrated in this paper, the web mapping application displays all criminal incidents that occurred within Washington D.C. between January 1 and December 31, 2017. The application allows the user to select either general crimes or narrow them down to specific offenses through the implementation of a dropdown box. Data is also grouped via marker clusters to display the number of incidents that occurred in a region depending on the zoom level. Finally, clicking on each data point displays a specific time and date that each incident occurred.

In general, we now have an interactive map of crime data in Washington D.C. There is a variety of crime data and other shapefiles through Open Data DC. This gives an idea of future aspects that could be added to create an even more comprehensive map of crime in D.C. Possible ideas for future functionalities to be added include:

- Adding more than 1 PostGIS query of crime data to Leaflet, allowing the user to select through different years of crime data
- Adding on layer control within the application to allow the user not just to select through different offenses, but maybe also the ward that crimes were committed in

## 6. REFERENCES

- [1] Taylor, D. 2018. DC Is America's 16th Most Dangerous City. Retrieved November 23, 2018 from <https://patch.com/district-columbia/washingtondc/dc-americas-16th-most-dangerous-city>
- [2] Open Data DC. 2018. Retrieved November 23, 2018 from <http://opendata.dc.gov/>
- [3] Open Data DC. Crime Incidents in 2017. Retrieved November 23, 2018 from <http://opendata.dc.gov/datasets/crime-incidents-in-2017?selectedAttribute=YBLOCK>
- [4] Carto. 2018. Carto – Location Intelligence Software. Retrieved November 23, 2018 from <https://carto.com/>
- [5] CodePen. 2018. Retrieved November 23, 2018 from <https://codepen.io/>
- [6] Leaflet. 2018. An open-source JavaScript library for mobile-friendly interactive maps. Retrieved November 23, 2018 from <https://leafletjs.com/>
- [7] Leaflet-providers preview. 2018. Retrieved November 23, 2018 from <https://leaflet-extras.github.io/leaflet-providers/preview/>
- [8] W3schools. 2018. HTML Color Picker. Retrieved November 23, 2018 from [https://www.w3schools.com/colors/colors\\_picker.asp](https://www.w3schools.com/colors/colors_picker.asp)
- [9] Carto. 2018. SQL API v2. Retrieved November 23, 2018 from <https://carto.com/developers/sql-api/>