Bike Counter

Data from NYCDOT Data Bike Counter

All imports for notebook contained in the below cell.

In [25]:

```
import pandas as pd  import numpy as np  import os
import glob
pd.options.display.max_rows = 400
import matplotlib.pyplot as plt
import seaborn as sns; sns.set(style="white", color_codes=True)
```

Creates a list of hles to be used in for loop

In [26]:

```
files                    =                    ['https://raw.githubusercontent.com/aaarista/Website/master/NYCDOT_20Bicyc
le_20Counts_20-_20East_20River_20Bridges/04%20April%202016%20Cyclist%20Numbers%20f      or%20Web.csv',
'https://raw.githubusercontent.com/aaarista/Website/master/NYCDOT_2
0Bicycle_20Counts_20-_20East_20River_20Bridges/05%20May%202016%20Cyclist%20Numbers
%20for%20Web.csv',
             'https://raw.githubusercontent.com/aaarista/Website/master/NYCDOT_20Bicycl
e_20Counts_20-_20East_20River_20Bridges/06%20June%202016%20Cyclist%20Numbers%20fo  r%20Web.csv',
'https://raw.githubusercontent.com/aaarista/Website/master/NYCDOT_20
Bicycle_20Counts_20-_20East_20River_20Bridges/07%20July%202016%20Cyclist%20Numbers
%20for%20Web.csv',
             'https://raw.githubusercontent.com/aaarista/Website/master/NYCDOT_20Bicycl
e_20Counts_20-_20East_20River_20Bridges/08%20August%202016%20Cyclist%20Numbers%20f    or%20Web.csv',
             'https://raw.githubusercontent.com/aaarista/Website/master/NYCDOT_20Bicycl
e_20Counts_20-_20East_20River_20Bridges/09%20September%202016%20Cyclist%20Numbers%
20for%20Web.csv',               'https://raw.githubusercontent.com/aaarista/Website/master/NYCDO
T_20Bicycle_20Counts_20-_20East_20River_20Bridges/10%20October%202016%20Cyclist%20
Numbers%20for%20Web.csv']
```

Creates empty dataframe and for loop to parse and read data from each hle as well as concatenate to one dataframe for further manipulation.

In [27]:

```
all_data = pd.DataFrame()
for f in glob.glob("/Users/macbookair/NYCDOT_20Bicycle_20Counts_20-_20East_20River_20Bridges/*"):
    dateparse = lambda x: pd.datetime.strptime(x, '%m-%d %M:%S') df = pd.read_csv(f, error_bad_lines=False,
        thousands=',') all_data = all_data.append(df,ignore_index=True, sort=True)
```

In [29]:

```
all_data.shape
```

Out[29]:

```
(320, 14)
```

Transforms the 'Date' column to datetime format

In [30]:

```
all_data['Date'] = pd.to_datetime(all_data['Date'], format = "%m/%d")
```

Sets the dataframe to the columns that will be used.

In [31]:

```
all_data = all_data[['Date', 'Brooklyn Bridge','Day', 'High Temp (°F)', 'Low Temp  (°F)', 'Manhattan Bridge', 'Williamsburg
Bridge', 'Precipitation', 'Queensboro Br  idge', 'Total']]
```

Repair the Date column year to 2016

There was no year given in the initial datasource. The default was 1900 which makes the data unclear. The correct year of the datasource is 2016. The following codes achieves this task:

In [32]:

```
all_data['Date'] = all_data['Date'].mask(all_data['Date'].dt.year == 1900,
                                         all_data['Date'] + pd.offsets.DateOffset(year=2016))

all_data.head()
```

	Date	Brooklyn Bridge	Day	High Temp (°F)	Low Temp (°F)	Manhattan Bridge	Williamsburg Bridge	Pre
0	2016-09-01	1608.0	Thursday	78.1	70.0	3012.0	4435.0	T
1	2016-09-02	3594.0	Friday	80.1	66.0	6657.0	7116.0	0.0
2	2016-09-03	2850.0	Saturday	73.9	68.0	7357.0	5115.0	0.0
3	2016-09-04	2871.0	Sunday	79.0	64.9	6949.0	4800.0	0.0
4	2016-09-05	2465.0	Monday	82.9	66.0	6248.0	4904.0	0.0

Set index to date

In [33]:

```
all_data.index = all_data['Date']
del all_data['Date']
```

Drops all rows that have Nan value in 'Total' column.

In [34]:

```
all_data.dropna(subset=['Total'], inplace=True)
```

To make the data more user-friendly the 'Date' column will be sorted ascending from the earliest date.

In [35]:

```
all_data.sort_values(by='Date').head()
```

Out[35]:

	Brooklyn Bridge	Day	High Temp (°F)	Low Temp (°F)	Manhattan Bridge	Williamsburg Bridge	Precipit
Date							
2016-04-01	1704.0	Friday	78.1	66.0	3126.0	4115.0	0.01
2016-04-02	827.0	Saturday	55.0	48.9	1646.0	2565.0	0.15
2016-04-03	526.0	Sunday	39.9	34.0	1232.0	1695.0	0.09
2016-04-04	521.0	Monday	44.1	33.1	1067.0	1440.0	0.47 (S)
2016-04-05	1416.0	Tuesday	42.1	26.1	2617.0	3081.0	0.00

In [36]:

```
all_data['Precipitation'] = all_data['Precipitation'].replace("T", 0)  all_data['Precipitation'] =
all_data['Precipitation'].replace("0.47 (S)", 0.47)
```

In [37]:

```
all_data['Precipitation'] = all_data['Precipitation'].convert_objects(convert_nume  ric=True)
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:1: FutureWarning:  convert_objects is deprecated. To
re-infer data dtypes for object columns, us  e Series.infer_objects()
For all other conversions use the data-type specific converters pd.to_datetim
e, pd.to_timedelta and pd.to_numeric.
  """Entry point for launching an IPython kernel.
```

Change datatype to float

In [38]:

```
all_data[['Manhattan Bridge', 'Williamsburg Bridge', 'Queensboro Bridge']].astype(  float).head()
```

Out[38]:

	Manhattan Bridge	Williamsburg Bridge	Queensboro Bridge
Date			
2016-09-01	3012.0	4435.0	3498.0
2016-09-02	6657.0	7116.0	5376.0
2016-09-03	7357.0	5115.0	3961.0
2016-09-04	6949.0	4800.0	3275.0
2016-09-05	6248.0	4904.0	3583.0

In [39]:

```
all_data.dtypes
```

Out[39]:

```
Brooklyn Bridge  Day               float64
High Temp (°F)                      object
Low Temp (°F)  Manhattan           float64
Bridge  Williamsburg               float64
Bridge  Precipitation              float64
Queensboro Bridge  Total           float64
dtype: object                      float64
                                   float64
                                   float64
```

*****The data is now tranformed correctly so that analysis can be done.**

Data Grouping and Preliminary Analysis

Verify the correct number of days for each month

In [40]:

```
all_data.resample('M').size()
```

Out[40]:

```
Date
2016-04-30    30
2016-05-31    31
2016-06-30    30
2016-07-31    31
2016-08-31    31
2016-09-30    30
2016-10-31    31
Freq: M, dtype: int64
```

Basic statistical description of 'all_data' dataframe.

In [41]:

```
all_data.describe().round()
```

Out[41]:

	Brooklyn Bridge	High Temp (°F)	Low Temp (°F)	Manhattan Bridge	Williamsburg Bridge	Precipitation	Qu 4
count	214.0	214.0	214.0	214.0	214.0	214.0	21 0 6
mean	3031.0	75.0	62.0	5052.0	6161.0	0.0	430
std	1134.0	13.0	12.0	1745.0	1911.0	0.0	129
min	504.0	40.0	26.0	997.0	1440.0	0.0	134
25%	2388.0	66.0	53.0	3713.0	4884.0	0.0	340
50%	3076.0	78.0	65.0	5132.0	6334.0	0.0	439
75%	3685.0	85.0	71.0	6610.0	7858.0	0.0	53
max	8264.0	96.0	82.0	9152.0	9148.0	2.0	63

Charts

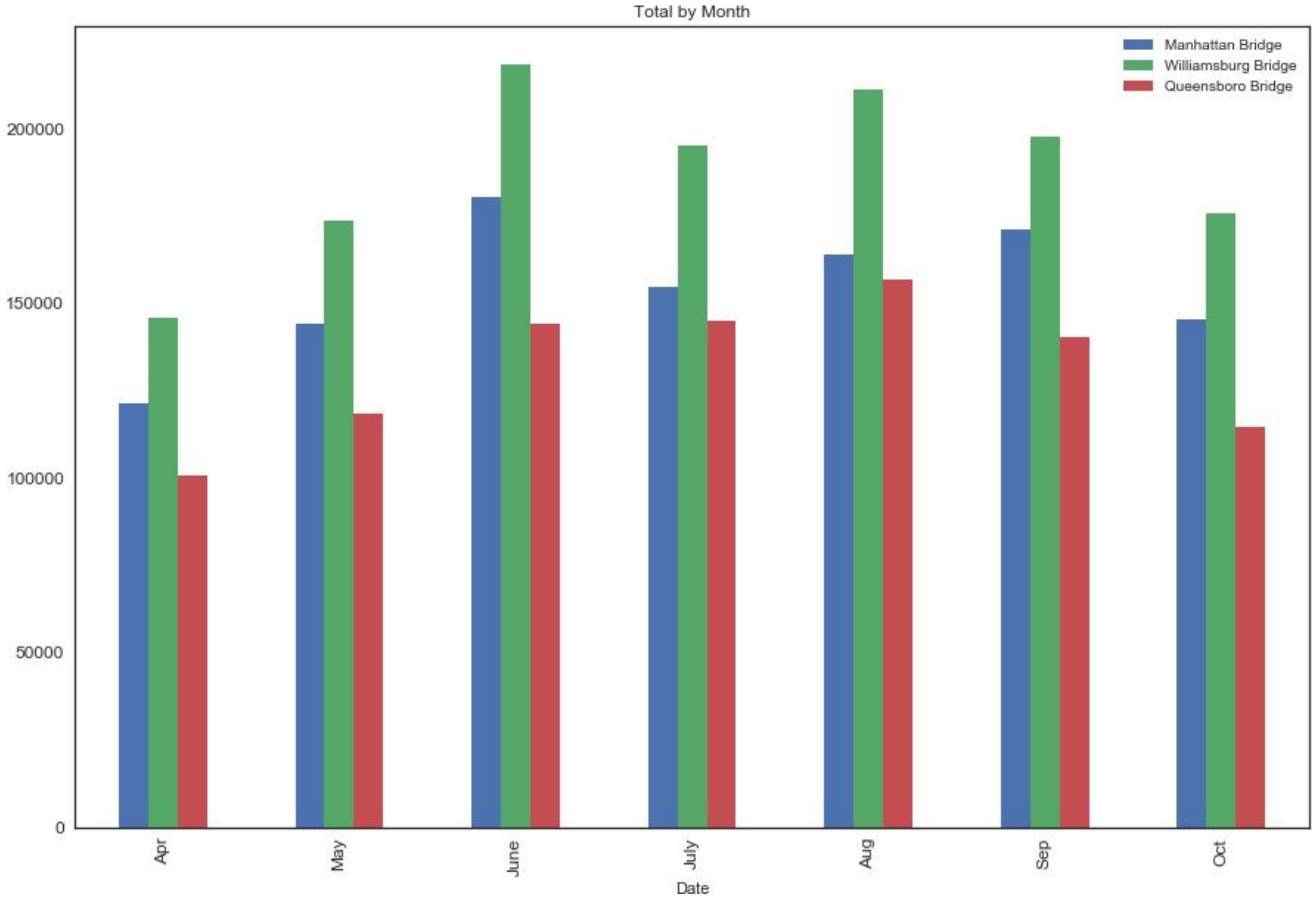
Total pedestrian cyclists aggregated by month

In [42]:

```
ttl_month = all_data[['Manhattan Bridge', 'Williamsburg Bridge', 'Queensboro Bridg  e']].resample('M').sum().round()
ax1 = ttl_month.plot(kind='bar', title ="Total by Month", figsize=(15, 10), legend=True, fontsize=12)
ax1.set_xticklabels(['Apr', 'May', 'June', 'July', 'Aug', 'Sep', 'Oct'])
```

Out[42]:

```
     [Text(0,0,'Apr'),
      Text(0,0,'May'),
 Text(0,0,'June'),
 Text(0,0,'July'),
      Text(0,0,'Aug'),
      Text(0,0,'Sep'),
      Text(0,0,'Oct')]
```


Total by Month
Manhattan Bridge
Williamsburg Bridge
Queensboro Bridge
200000
150000
100000
50000
0
Apr
May
June
July
Aug
Sep
Oct
Date

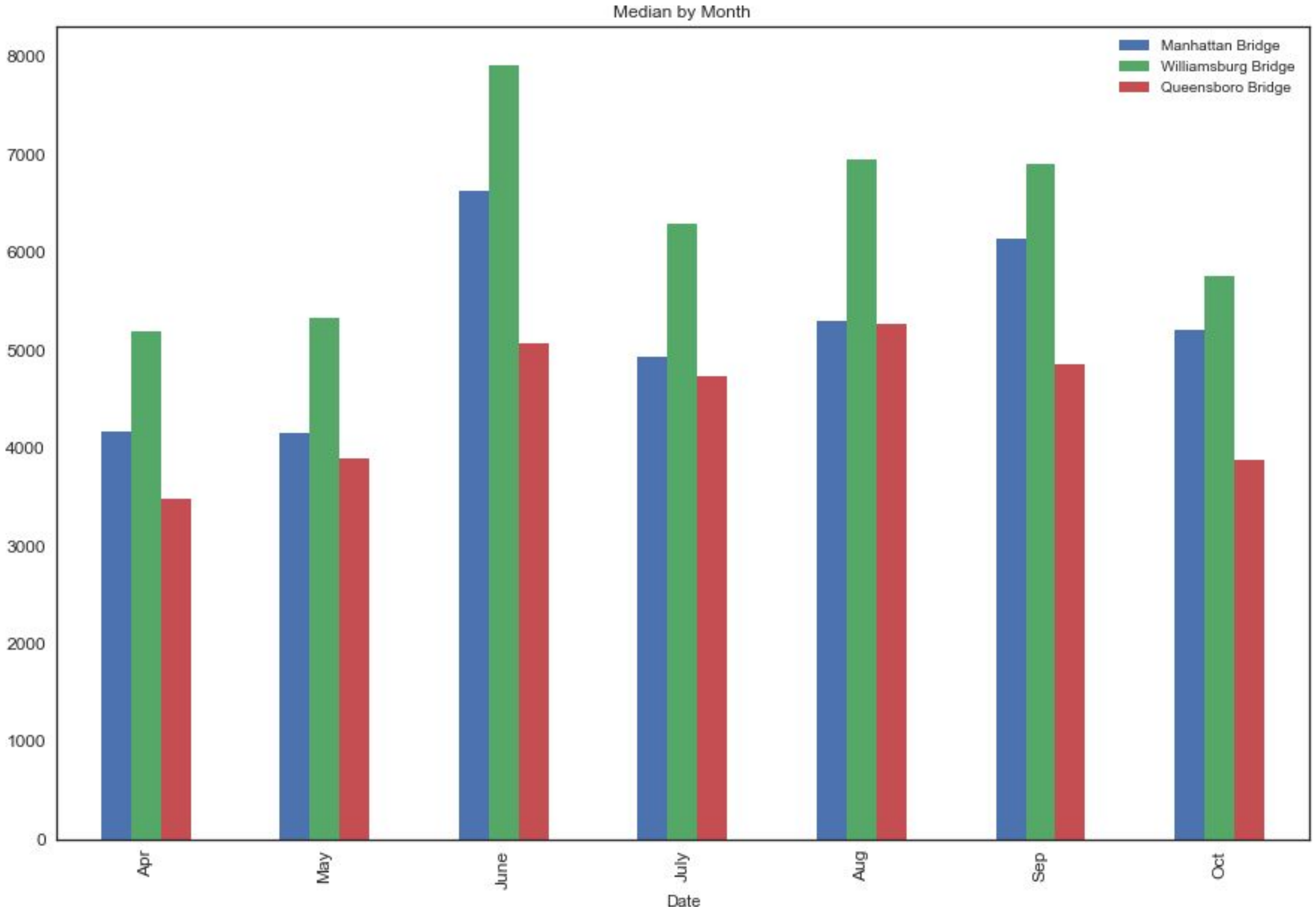
Median monthly pedestrian cyclists

In [43]:

```
med_month = all_data[['Manhattan Bridge', 'Williamsburg Bridge', 'Queensboro Bridg e']].resample('M').median().round()
ax2 = med_month.plot(kind='bar', title ="Median by Month", figsize=(15, 10), legend=True, fontsize=12)
ax2.set_xticklabels(['Apr', 'May', 'June', 'July', 'Aug', 'Sep', 'Oct'])
```

Out[43]:

```
     [Text(0,0,'Apr'),
      Text(0,0,'May'),
 Text(0,0,'June'),
 Text(0,0,'July'),
      Text(0,0,'Aug'),
      Text(0,0,'Sep'),
      Text(0,0,'Oct')]
```

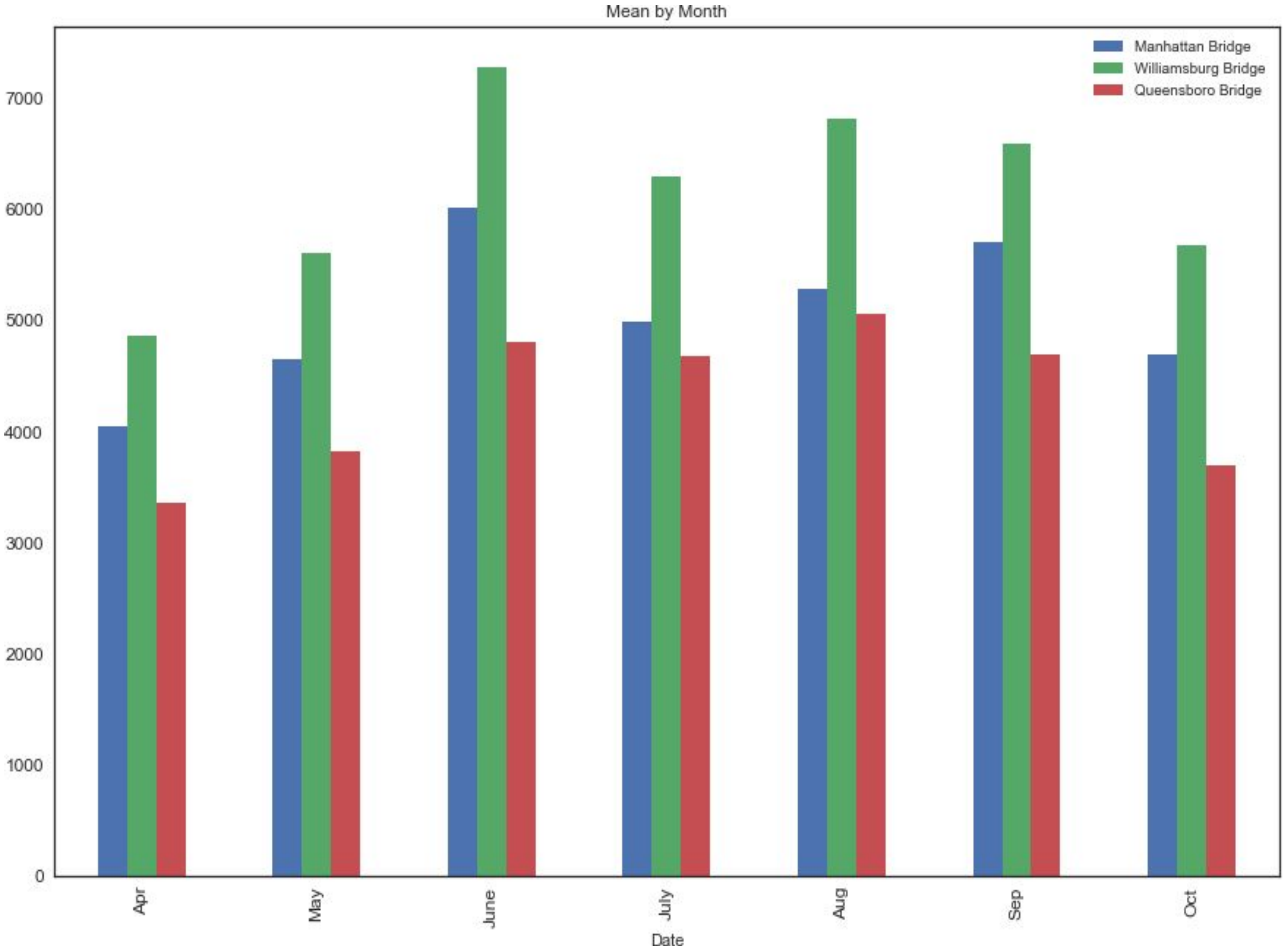


Median by Month
Manhattan Bridge
Williamsburg Bridge
Queensboro Bridge
8000
7000
6000
5000
4000
3000
2000
1000
0
Apr
May
June
July
Aug
Sep
Oct
Date

Mean pedestrian cyclists by month

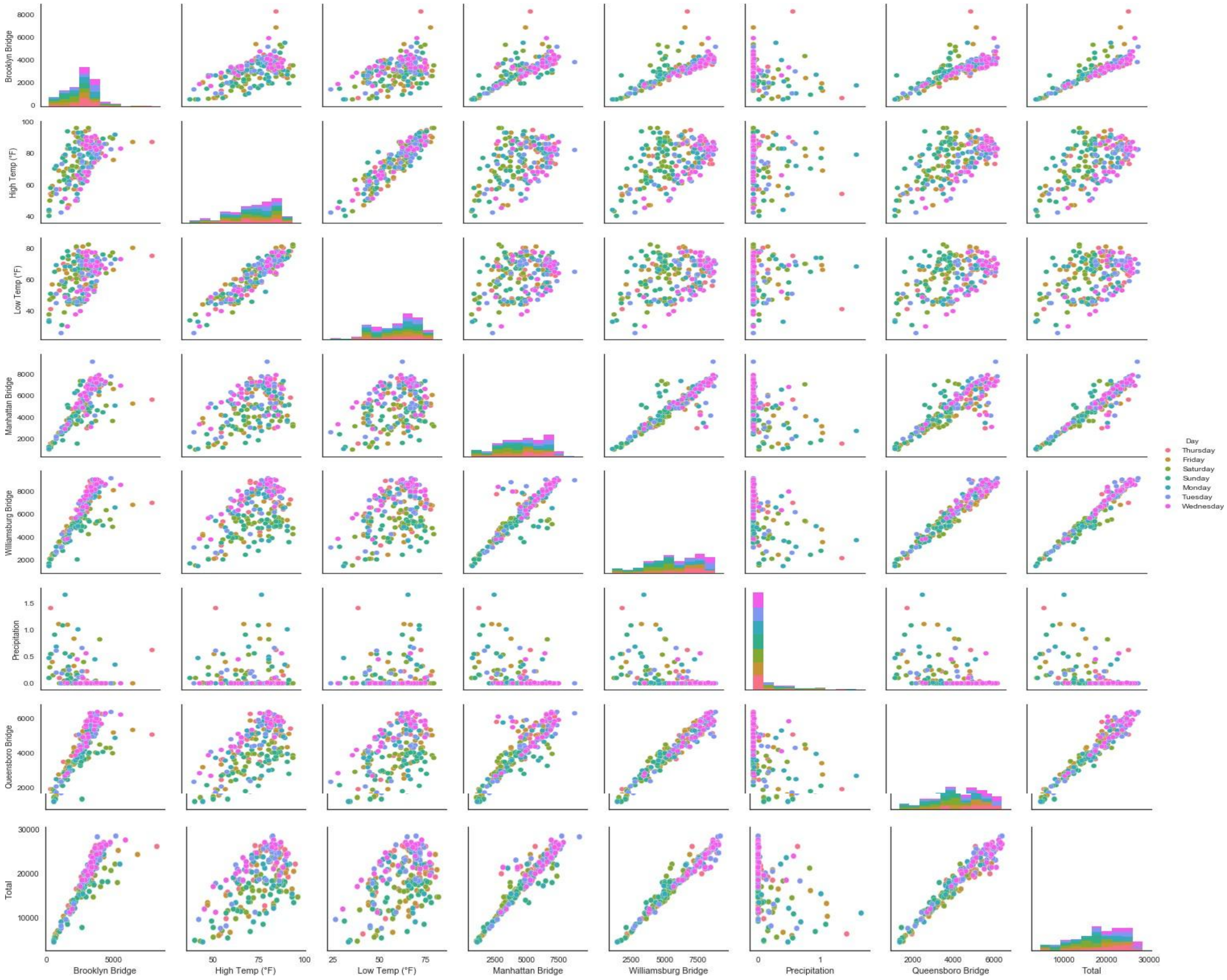
```
mean_month = all_data[['Manhattan Bridge', 'Williamsburg Bridge', 'Queensboro Brid  ge']].resample('M').mean().round()
ax3 = mean_month.plot(kind='bar', title ="Mean by Month", figsize=(15, 10), legend=True, fontsize=12)
ax3.set_xticklabels(['Apr', 'May', 'June', 'July', 'Aug', 'Sep', 'Oct'])
```

```
Out[44]:          [Text(0,0,'Apr'),
                   Text(0,0,'May'),
               Text(0,0,'June'),
               Text(0,0,'July'),
                   Text(0,0,'Aug'),
                   Text(0,0,'Sep'),
                   Text(0,0,'Oct')]
```


Mean by Month
Manhattan Bridge
Williamsburg Bridge
Queensboro Bridge
7000
6000
5000
4000
3000
2000
1000
0
Apr
May
June
July
Aug
Sep
Oct
Date

Seaborn pairplot to show temperature and precipitation impact on cyclists

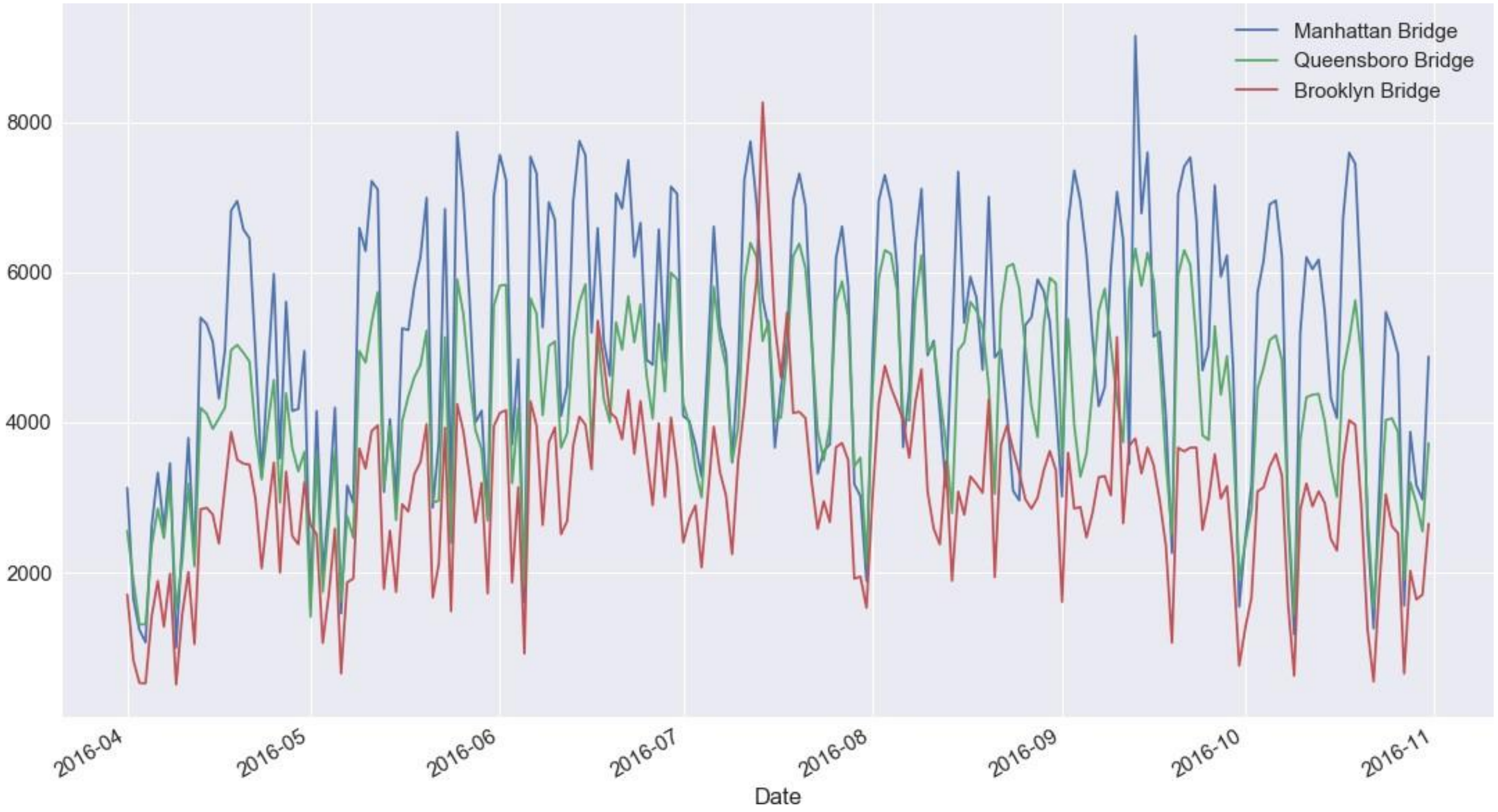
```
pp = sns.pairplot(all_data, hue='Day')  sns.set(font_scale=1.5)
```


Brooklyn Bridge
0
2000
4000
6000
8000
High Temp (°F)
40
60
80
100
Low Temp (°F)
40
60
80
Manhattan Bridge
2000
4000
6000
8000
Williamsburg Bridge
2000
4000
6000
8000
Precipitation
0.0
0.5
1.0
1.5
Queensboro Bridge
2000
4000
6000
Total
10000
20000
30000
Day
Thursday
Friday
Saturday
Sunday
Monday
Tuesday
Wednesday
0
5000
Brooklyn Bridge
50
75
100
High Temp (°F)
25
50
75
Low Temp (°F)
2500
5000
7500
Manhattan Bridge
2500
5000
7500
Williamsburg Bridge
0
1
Precipitation
2000
4000
6000
Queensboro Bridge
10000
20000
30000
Total

Time Series Analysis

In [46]:

```
grp_line = all_data[['Manhattan Bridge', 'Queensboro Bridge', 'Brooklyn Bridge']]  pp = grp_line.plot()
fig = plt.gcf()
fig.set_size_inches(18.5, 10.5)
```


Manhattan Bridge
Queensboro Bridge
Brooklyn Bridge
Date