09. Geographic Plots

Bokeh Tutorial

(http://bokeh.pydata.org/)

In [1]:
```
from bokeh.io import output_notebook, show
output_notebook()
```

(https://bokeh.pydata.org) BokehJS 0.12.14 successfully loaded.

It is often useful to be able to relate datasets with their real-world context. You can plot geographic data just like any other type of data, as in the Texas Unemployment example (https://bokeh.pydata.org/en/latest/docs/gallery/texas.html), but Bokeh also Bokeh provides several specialized mechanisms for plotting data in geographic coordinates:

- GMapPlot (https://bokeh.pydata.org/en/latest/docs/user_guide/geo.html#google-maps-support): Bokeh Plots on top of Google Maps
- TileSource (https://bokeh.pydata.org/en/latest/docs/user_guide/geo.html#tile-providers), especially WMTSTileSource: allows data to be overlaid on data from any map tile server, including Google Maps (http://maps.google.com), Stamen (http://maps.stamen.com), MapQuest (https://www.mapquest.com/), OpenStreetMap (https://www.openstreetmap.org), ESRI (http://www.esri.com), and custom servers.
- GeoJSONDataSource (https://bokeh.pydata.org/en/dev/docs/user_guide/geo.html#geojson-datasource): Allows reading data in GeoJSON (http://geojson.org/) format for use with Bokeh plots and glyphs, similar to `ColumnDataSource`.

WMTS Tile Source

WTMS is the most common web standard for tiled map data, i.e. maps supplied as standard-sized image patches from which the overall map can be constructed at a given zoom level. WTMS uses Web Mercator format, measuring distances from Greenwich, England as meters north and meters west, which is easy to compute but does distort the global shape.

First let's create an empty Bokeh plot covering the USA, with bounds specified in meters:

In [2]:
```
from bokeh.plotting import figure
from bokeh.models import WMTSTileSource

# web mercator coordinates
USA = x_range,y_range = ((-13884029,-7453304), (2698291,6455972))

p = figure(tools='pan, wheel_zoom', x_range=x_range, y_range=y_range)
p.axis.visible = False
```

A few WTMS tile sources are already defined in `bokeh.tile_providers`, but here we'll show how to specify the interface using a format string showing Bokeh how to request a tile with the required zoom, x, and y values from a given tile provider:

In [3]:
```
url = 'http://a.basemaps.cartocdn.com/dark_all/{Z}/{X}/{Y}.png'
attribution = "Tiles by Carto, under CC BY 3.0. Data by OSM, under ODbL"

p.add_tile(WMTSTileSource(url=url, attribution=attribution))
```

Out[3]: **TileRenderer**(id = 'e3cbe843-ea04-4a47-8f90-69e67e4be6fe', …)

If you show the figure, you can then use the wheel zoom and pan tools to navigate over any zoom level, and Bokeh will request the appropriate tiles from the server and insert them at the correct locations in the plot:

In [4]:
```
show(p)
```

UNITED STATES OF AMERICA
MEXICO
CUBA
GUATEMALA
Tiles by Carto, under CC BY 3.0. Data by OSM, under ODbL

(htt

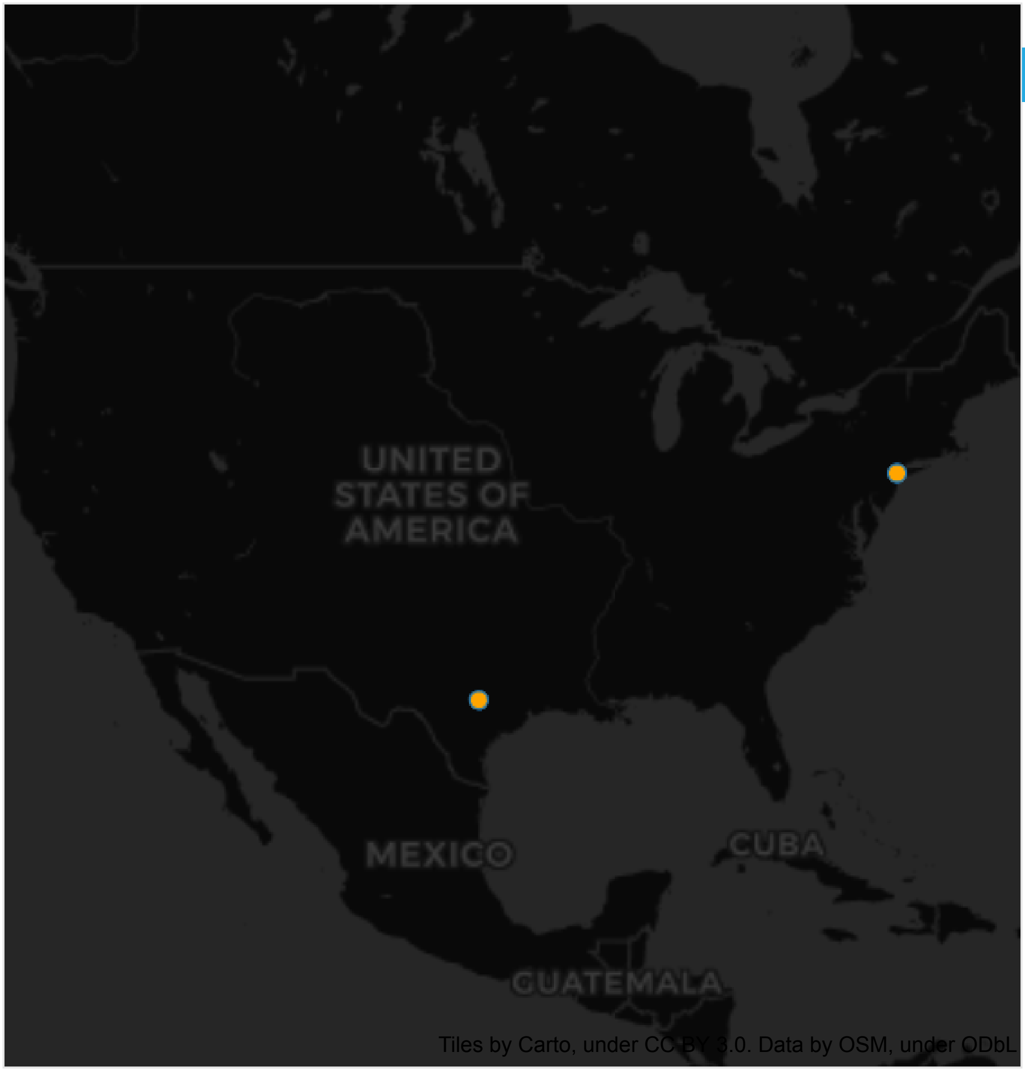
That's all it takes to put map data into your plot! Of course, you'll usually want to show other data as well, or you could just use the tile server's own web address. You can now add anything you would normally use in a Bokeh plot, as long as you can obtain coordinates for it in Web Mercator format. For example:

In [5]:
```
import pandas as pd
import numpy as np

def wgs84_to_web_mercator(df, lon="lon", lat="lat"):
    """Converts decimal longitude/latitude to Web Mercator format"""
    k = 6378137
    df["x"] = df[lon] * (k * np.pi/180.0)
    df["y"] = np.log(np.tan((90 + df[lat]) * np.pi/360.0)) * k
    return df

df = pd.DataFrame(dict(name=["Austin", "NYC"], lon=[-97.7431,-74.0059], lat=[30.2672,40.7128
wgs84_to_web_mercator(df)
```

Out[5]:

	lat	lon	name	x	y
0	30.2672	-97.7431	Austin	-1.088071e+07	3.537942e+06
1	40.7128	-74.0059	NYC	-8.238299e+06	4.970072e+06

In [6]:
```
p.circle(x=df['x'], y=df['y'], fill_color='orange', size=10)
show(p)
```



(htt