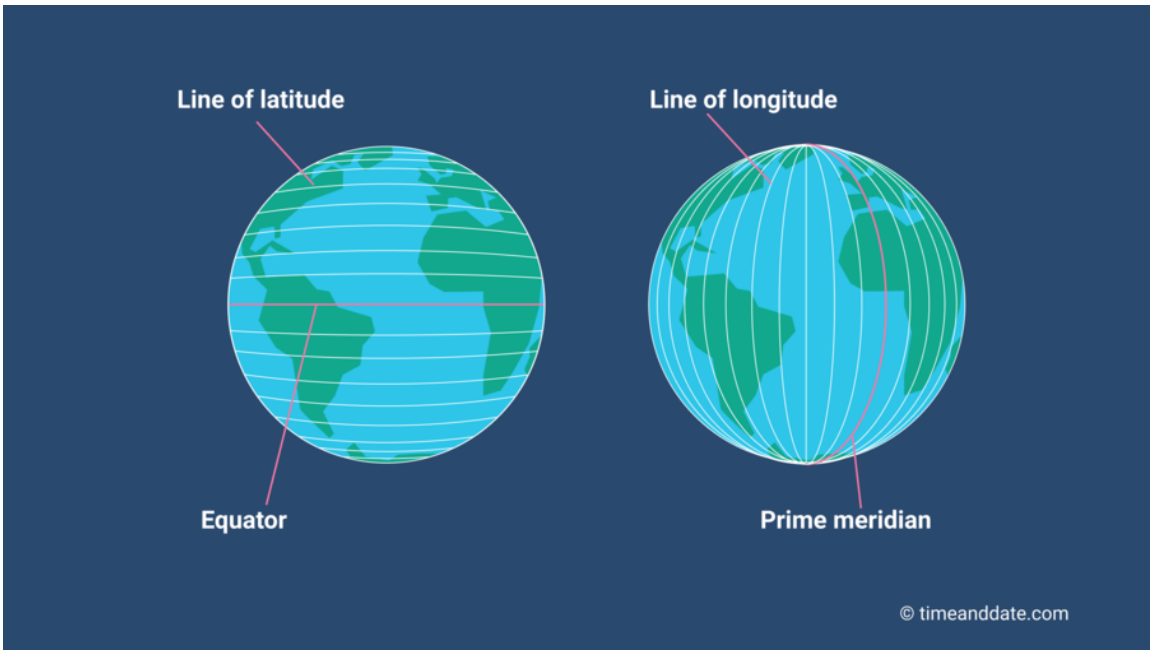
Bokeh (https://docs.bokeh.org/en/latest/docs/user_

Bokeh is a Python interactive visualization library that targets modern web browsers for presentation. It can be used to generate interactive maps with features such as zooming, panning, and hovering.

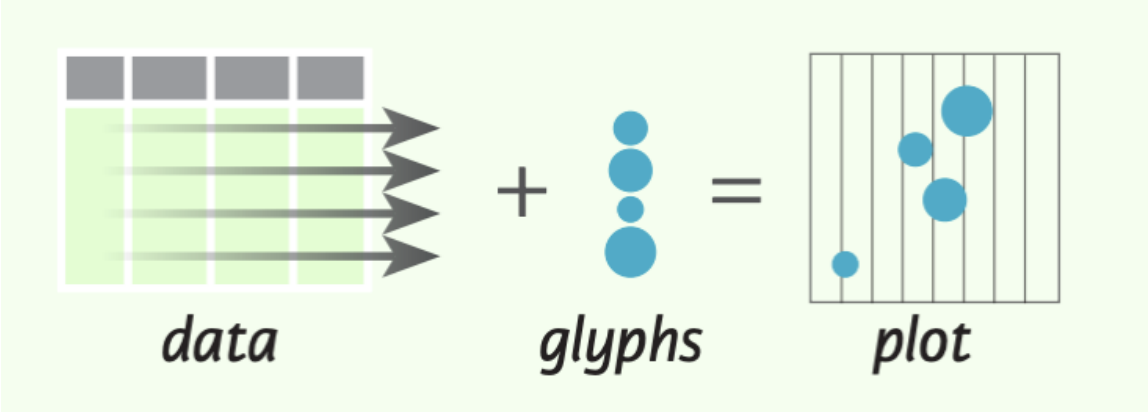
Interactive maps are used to visualize the data based on the geo-location category. any large dataset which contains a lot of geo-location data like cities, states, countries, etc can be plotted easily. bokeh is an open-source package, which uses the Bokeh visualization tool. It gives a flexible declarative interface for dynamic web-based visualizations as well as an interactive dashboard.



Reference: https://www.timeanddate.com/geography/longitude-latitude.html

The Python interactive visualization library Bokeh enables high-performance visual presentation of large datasets in modern web browsers.

Bokeh's mid-level general-purpose bokeh. plotting interface is centered around two main components: data and glyphs.



Bokeh is a data visualization library in Python that provides high-performance interactive charts and plots. Bokeh output can be obtained in various mediums like notebook, html and server. It is possible to embed bokeh plots in Django and flask apps.

Bokeh provides two visualization interfaces to users:

1. bokeh.models : A low level interface that provides high flexibility to application developers.
2. bokeh.plotting : A high level interface for creating visual glyphs.

Bokeh is compatible with several XYZ tile services that use the Web Mercator projection. Bokeh uses the xyzservices(https://xyzservices.readthedocs.org) library to take care of the tile sources and their attributions.

To add these to a plot, use the method add_tile(). You can pass any name xyzservices may recognize. The retina keyword can control the resolution of tiles.

Notice that passing **x_axis_type="mercator"** and **y_axis_type="mercator"** to figure generates axes with latitude and longitude labels, instead of raw Web Mercator coordinates.

In [ ]:

Example 1: In this example, we will create an exemplary dataset and then plot a Map using that Coordinates.

Dataset is:

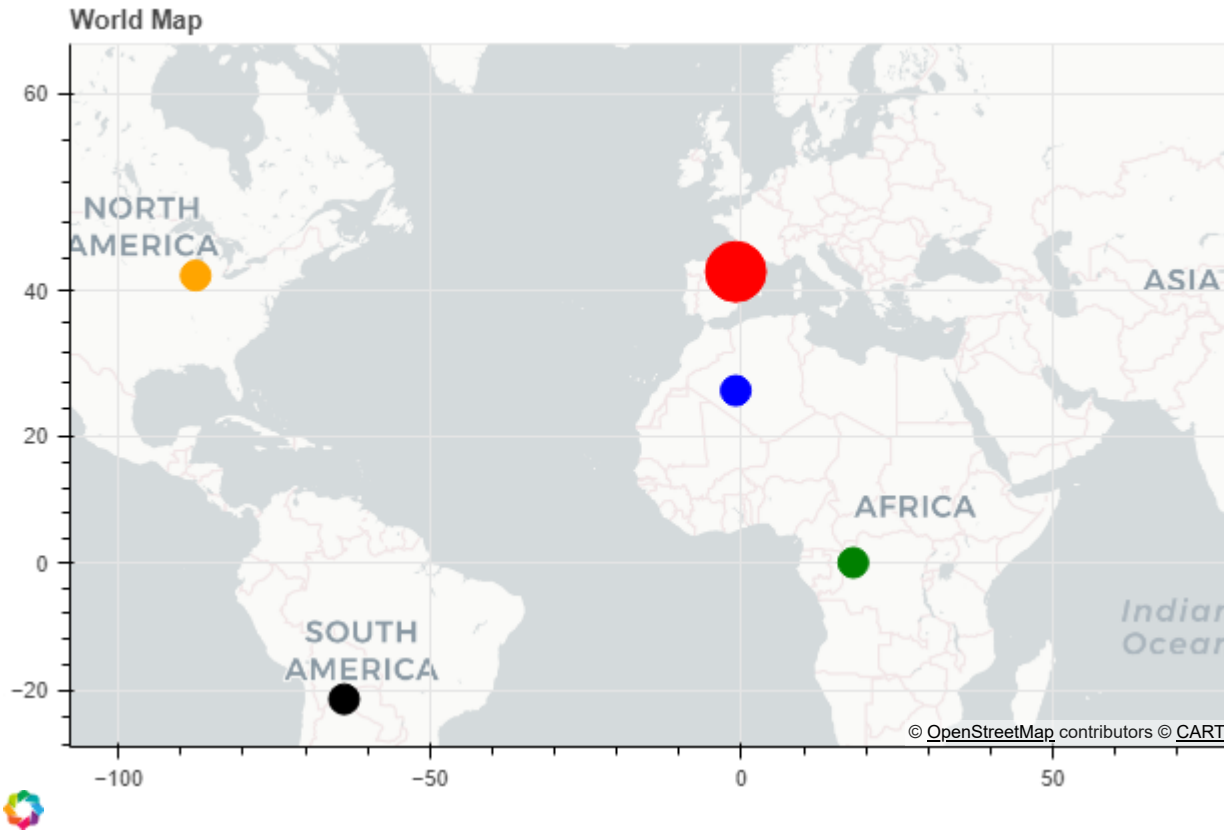
X-Coordinate (Latitude)	Y-Coordinate (Longitude)
-100833	5211172
-100833	3086289
-9754910	5142738
1999900	12738
-7100000	-2425502

How to do this visualization?

1. Import Library.
2. Initialize the tile provider.
3. Provide the data needed to be displayed to the tuple.
4. Pass height, width, and ranged x,y coordinates to figure(width, height) function.
5. Add title.
6. Provide the required coordinates to the circle function.
7. Mention circle size and color.
8. After circling display using show() function.

In [86]:
```
# Python program to make an
# interactive map using bokeh library

from bokeh.plotting import show, output_notebook, figure

# Provide the data tuple needed to be display while hovering.
tooltips = ("Welcome")

# Creating Map object
m = figure(title='World Map',
           width=650,
           height=400,
           x_range=(-12000000, 9000000),
           y_range=(-1000000, 7000000),
           x_axis_type='mercator',
           y_axis_type='mercator',
           tooltips=tooltips)

# # Adding tile
m.add_tile("CartoDB Positron", retina=True)
```

```
# # Circling the coordinates.
m.circle(x=-100833, y=5211172, size=30, color='red')
m.circle(x=-100833, y=3086289, size=15, color='blue')
m.circle(x=-9754910, y=5142738, size=15, color='orange')
m.circle(x=1999900, y=12738, size=15, color='green')
m.circle(x=-7100000, y=-2425502, size=15, color='black')

# # Displaying the Map using show function
show(m)
output_notebook()
```



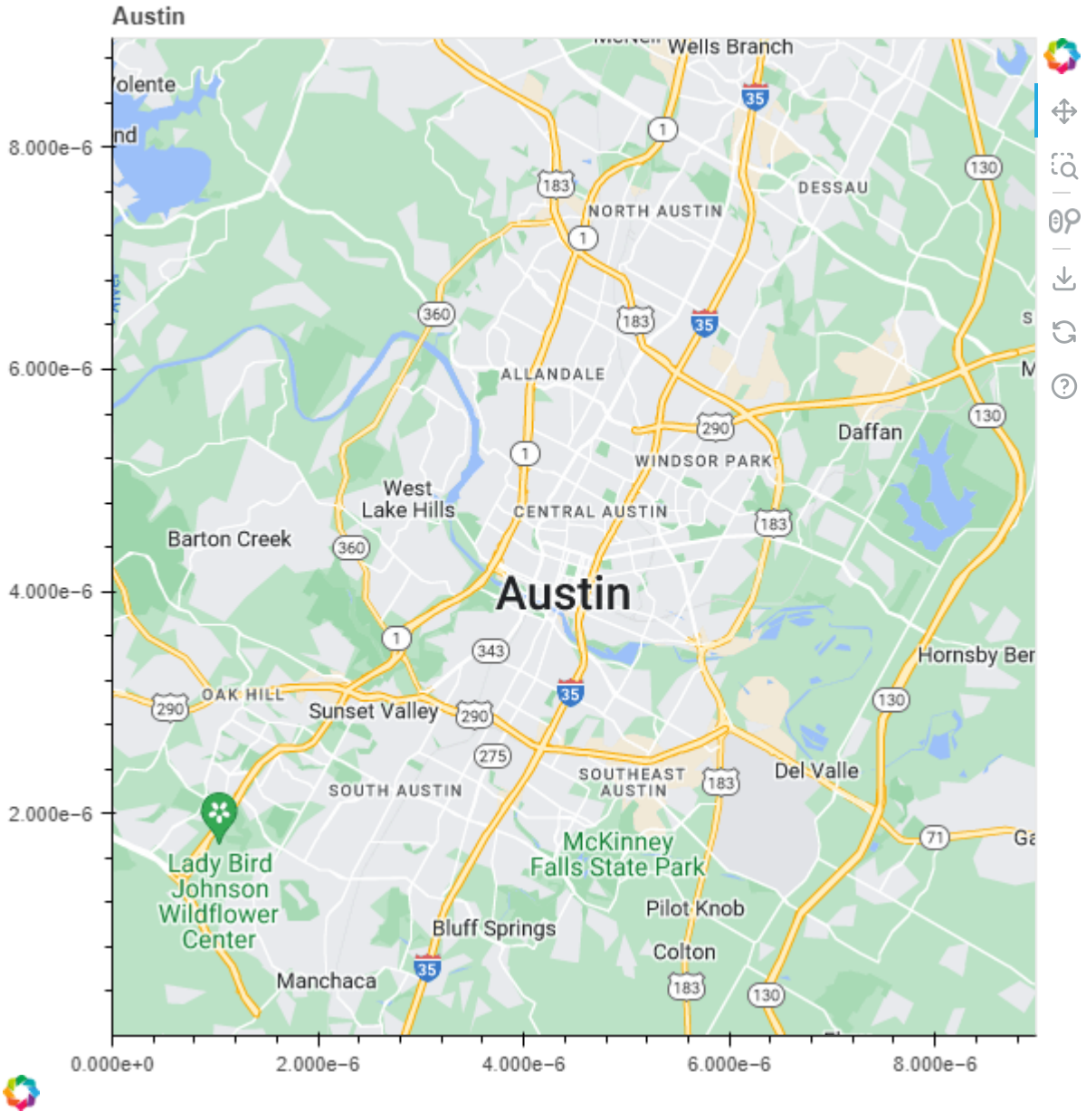
Loading BokehJS ...

Google Maps

To plot glyphs over a Google Map, use the function gmap(). For the function to work, you must pass it a Google API Key and configure the Google Map underlay GMapOptions. The Google API Key will be stored in the Bokeh Document JSON.

In [83]:
```
from bokeh.models import ColumnDataSource, GMapOptions
from bokeh.plotting import gmap, show, output_notebook

map_options = GMapOptions(lat=30.2861, lng=-97.7394, map_type="roadmap", zoom=11)

# For GMaps to function, Google requires you obtain and enable an API key:
#
#     https://developers.google.com/maps/documentation/javascript/get-api-key
#
```

```
# Replace the value below with your personal API key:
p = gmap("GOOGLE_API_KEY", map_options, title="Austin")

source = ColumnDataSource(
    data=dict(lat=[ 30.29,  30.20,  30.29],
              lon=[-97.70, -97.74, -97.78]),
)

p.circle(x="lon", y="lat", size=15, fill_color="blue", fill_alpha=0.8, source=sourc

show(p)
output_notebook()
```



Loading BokehJS ...

GeoJSON data

GeoJSON is a popular open standard for representing geographical features with JSON. It describes points, lines, and polygons (called Patches in Bokeh) as a collection of features. Each feature can also have a set of properties.

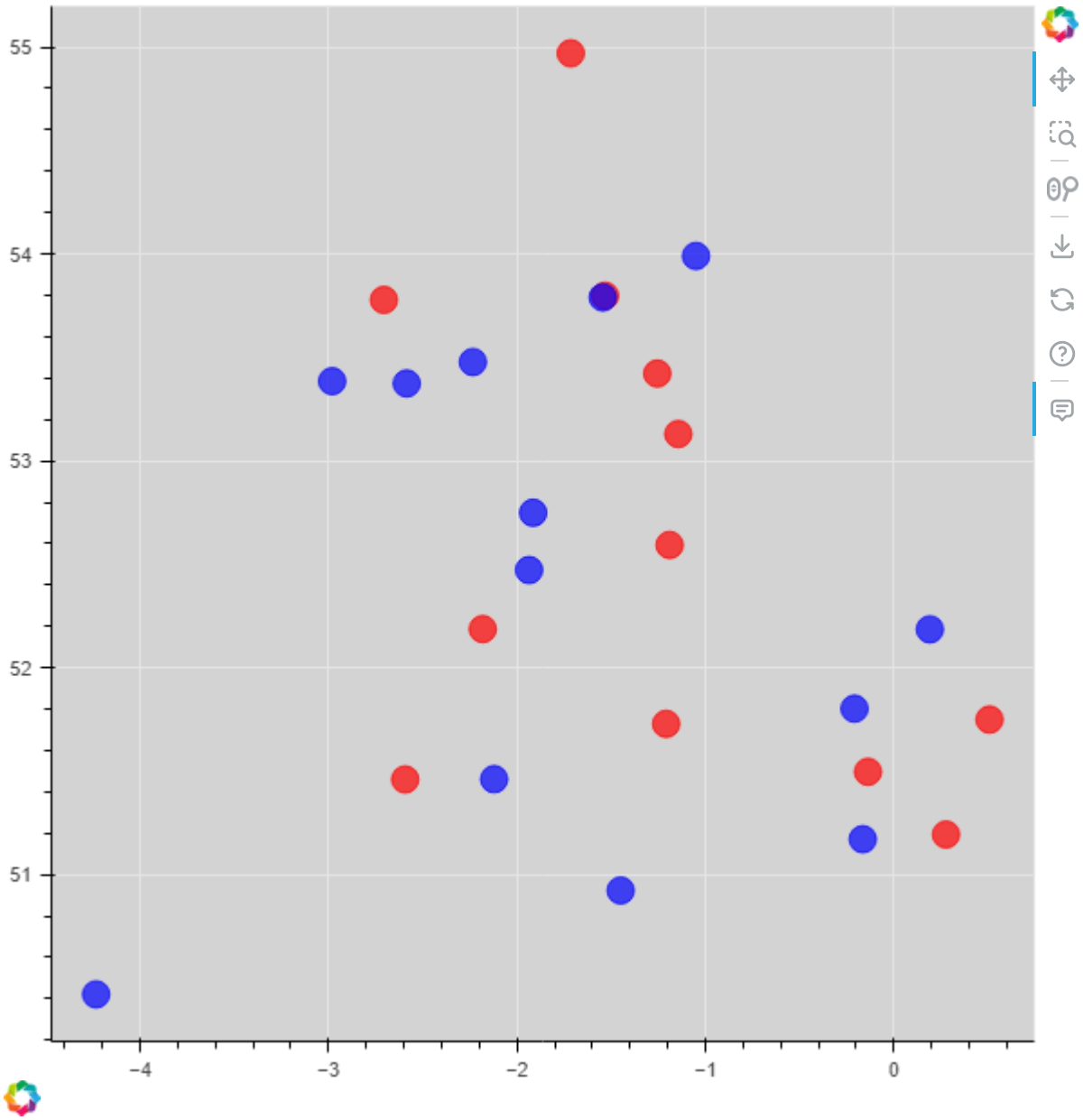
Bokeh's GeoJSONDataSource can be used almost seamlessly in place of Bokeh's ColumnDataSource.

All the list of datasets are available here:
https://docs.bokeh.org/en/latest/docs/reference/sampledata.html#

Dataset used in below example is available here:

- https://docs.bokeh.org/en/latest/docs/reference/sampledata.html#module-bokeh.sampledata.sample_geojson

In [82]:
```
import json

from bokeh.models import GeoJSONDataSource
from bokeh.plotting import figure, show, output_notebook
from bokeh.sampledata.sample_geojson import geojson

data = json.loads(geojson)

for i in range(len(data['features'])):
    data['features'][i]['properties']['Color'] = ['blue', 'red'][i%2]

geo_source = GeoJSONDataSource(geojson=json.dumps(data))

TOOLTIPS = [('Organisation', '@OrganisationName')]

p = figure(background_fill_color="lightgrey", tooltips=TOOLTIPS)

p.circle(x='x', y='y', size=15, color='Color', alpha=0.7, source=geo_source)

show(p)
output_notebook()
```



BokehJS 3.2.1 successfully loaded.

In [ ]: