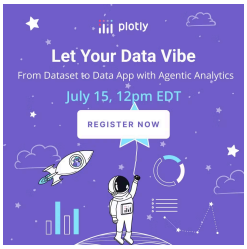
Tile Map Layers in Python

How to make tile-based maps in Python with various base layers.



Tile Maps vs Outline Maps

Plotly supports two different kinds of maps:

- Tile-based maps (https://en.wikipedia.org/wiki/Tiled_web_map)

If your figure is created with a px.scatter_map, px_scatter_mapbox, px.line_map, px.line_mapbox, px.choropleth_map, px.choropleth_mapbox, px.density_map, or px.density_mapbox function or otherwise contains one or more traces of type go.Scattermap, go.Scattermapbox, go.Choroplethmap, go.Choroplethmapbox, go.Densitymap, or go.Densitymapbox, the layout.map or layout.mapbox object in your figure contains configuration information for the map itself.

- **Outline-based maps**

Geo maps are outline-based maps. If your figure is created with a px.scatter_geo, px.line_geo or px.choropleth function or otherwise contains one or more traces of type go.Scattergeo or go.Choropleth, the layout.geo object in your figure contains configuration information for the map itself.

> This page documents tile-based maps, and the Geo map documentation (/python/map-configuration/) describes how to configure outline-based maps.

Tile Map Renderers

Tile-based traces in Plotly use Maplibre or Mapbox.

Maplibre-based traces (new in 5.24) are ones generated in Plotly Express using px.scatter_map, px.line_map, px.choropleth_map, px.density_map, or Graph Objects using go.Scattermap, go.Choroplethmap, or go.Densitymap.

Mapbox-based traces are suffixed with mapbox, for example go.Scattermapbox. These are deprecated as of version 5.24 and we recommend using the Maplibre-based traces.

Maplibre

New in 5.24

Maplibre-based tile maps have three different types of layers:

- layout.map.style defines the lowest layers of the map, also known as the "base map".
- The various traces in data are by default rendered above the base map (although this can be controlled via the below attribute).
- layout.map.layers is an array that defines more layers that are by default rendered above the traces in data (although this can also be controlled via the below attribute.

Base Maps in layout.map.style.

The accepted values for layout.map.style are one of:

- "basic"
- "carto-darkmatter"
- "carto-darkmatter-nolabels"
- "carto-positron"
- "carto-positron-nolabels"
- "carto-voyager"
- "carto-voyager-nolabels"
- "dark"
- "light"
- "open-street-map"
- "outdoors"
- "satellite"
- "satellite-streets"
- "streets"
- "white-bg" - an empty white canvas which results in no external HTTP requests

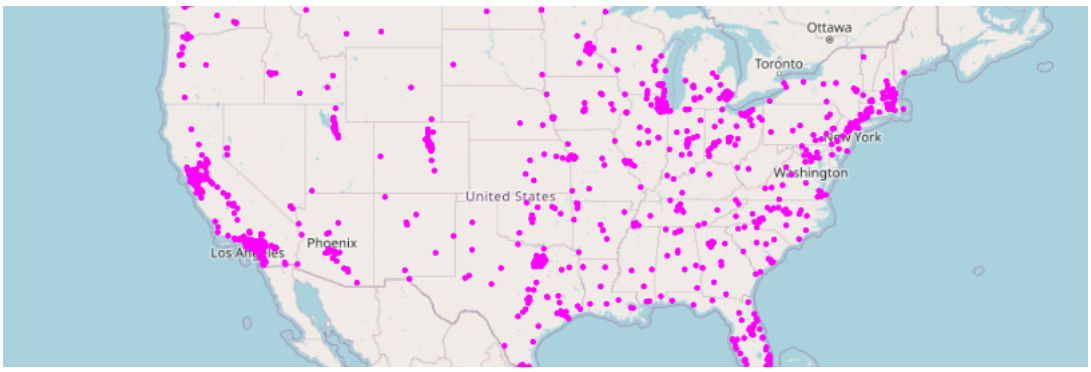
. For example: https://tiles.stadiamaps.com/styles/stamen_watercolor.json?api_key=YOUR-API-KEY
maps.com/styles/stamen_watercolor.json?api_key=YOUR-API-KEY)

as defined at https://maplibre.org/maplibre-style-spec/ (https://maplibre.org/maplibre-style-spec/)

OpenStreetMap tiles

Here is a simple map rendered with OpenStreetMaps tiles.

```
import pandas as pd
us_cities = pd.read_csv("https://raw.githubusercontent.com/plotly/datasets/master/us-cities-top-1k.csv")

import plotly.express as px

fig = px.scatter_map(us_cities, lat="lat", lon="lon", hover_name="City", hover_data=["State", "Population"],
                        color_discrete_sequence=["fuchsia"], zoom=3, height=300)
fig.update_layout(map_style="open-street-map")
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```



Using layout.map.layers to Specify a Base Map

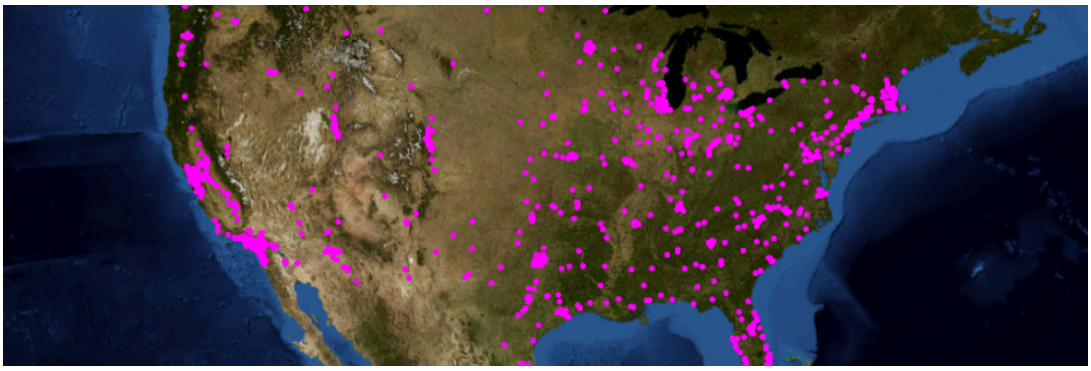
If you have access to your own private tile servers, or wish to use a tile server not included in the list above, the recommended approach is to set layout.map.style to "white-bg" and to use layout.map.layers with below to specify a custom base map.

> If you omit the below attribute when using this approach, your data will likely be hidden by fully-opaque raster tiles!

Base Tiles from the USGS: no token needed

Here is an example of a map which uses a public USGS imagery map, specified in layout.map.layers, and which is rendered *below* the data layer.

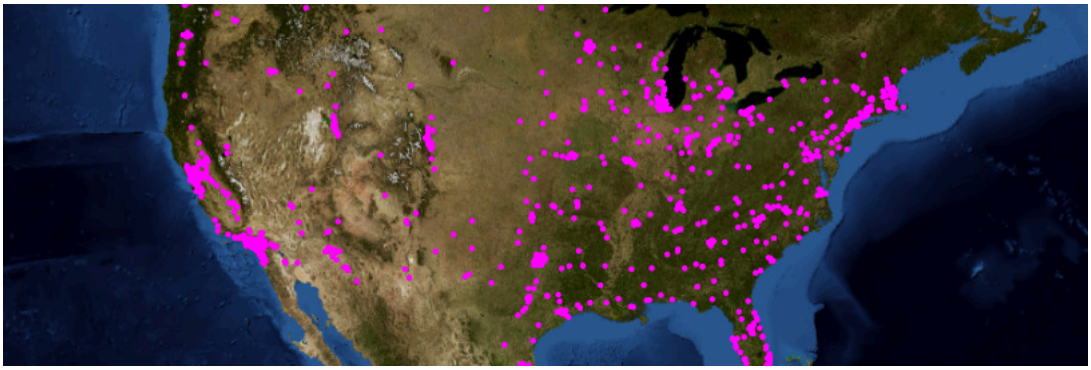
```
import pandas as pd
us_cities = pd.read_csv("https://raw.githubusercontent.com/plotly/datasets/master/us-cities-top-1k.csv")

import plotly.express as px

fig = px.scatter_map(us_cities, lat="lat", lon="lon", hover_name="City", hover_data=["State", "Population"],
                        color_discrete_sequence=["fuchsia"], zoom=3, height=300)
fig.update_layout(
    map_style="white-bg",
    map_layers=[
        {
            "below": 'traces',
            "sourcetype": "raster",
            "sourceattribution": "United States Geological Survey",
            "source": [
                "https://basemap.nationalmap.gov/arcgis/rest/services/USGSImageryOnly/MapServer/tile/{z}/{y}/{x}"
            ]
        }
      ])
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```



Base Tiles from the USGS, radar overlay from Environment Canada

Here is the same example, with in addition, a WMS layer from Environment Canada which displays near-real-time radar imagery in partly-transparent raster tiles, rendered above the go.Scattermap trace, as is the default:

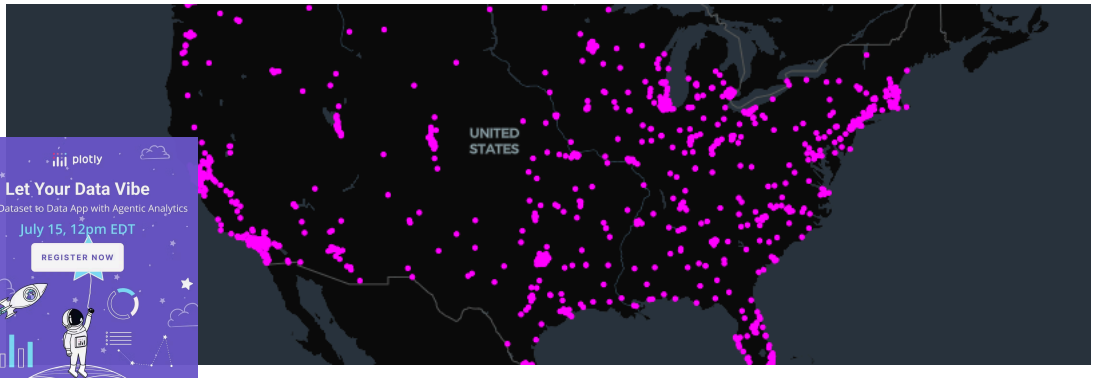
```
import pandas as pd
us_cities = pd.read_csv("https://raw.githubusercontent.com/plotly/datasets/master/us-cities-top-1k.csv")

import plotly.express as px

fig = px.scatter_map(us_cities, lat="lat", lon="lon", hover_name="City", hover_data=["State", "Population"],
                        color_discrete_sequence=["fuchsia"], zoom=3, height=300)
fig.update_layout(
    map_style="white-bg",
    map_layers=[
        {
            "below": 'traces',
            "sourcetype": "raster",
            "sourceattribution": "United States Geological Survey",
            "source": [
                "https://basemap.nationalmap.gov/arcgis/rest/services/USGSImageryOnly/MapServer/tile/{z}/{y}/{x}"
            ]
        },
        {
            "sourcetype": "raster",
            "sourceattribution": "Government of Canada",
            "source": ["https://geo.weather.gc.ca/geomet/?"
                       "SERVICE=WMS&VERSION=1.3.0&REQUEST=GetMap&BBOX={bbox-epsg-3857}&CRS=EPSG:3857"
                       "&WIDTH=1000&HEIGHT=1000&LAYERS=RADAR_1KM_RDBR&TILED=true&FORMAT=image/png"],
        }
      ])
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```



Dark tiles example

Here is a map rendered with the "dark" style.

```
import pandas as pd
us_cities = pd.read_csv("https://raw.githubusercontent.com/plotly/datasets/master/us-cities-top-1k.csv")

import plotly.express as px

fig = px.scatter_map(us_cities, lat="lat", lon="lon", hover_name="City", hover_data=["State", "Population"],
                        color_discrete_sequence=["fuchsia"], zoom=3, height=300)
fig.update_layout(map_style="dark")
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```



Stamen Watercolor using a Custom Style URL

Here's an example of using a custom style URL that points to the Stadia Maps (https://docs.stadiamaps.com/map-styles/stamen-watercolor) service to use the stamen_watercolor base map.

```
import pandas as pd
quakes = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/earthquakes-23k.csv')

import plotly.graph_objects as go
fig = go.Figure(go.Densitymap(lat=quakes.Latitude, lon=quakes.Longitude, z=quakes.Magnitude,
                              radius=10))
fig.update_layout(map_style="https://tiles.stadiamaps.com/styles/stamen_watercolor.json?api_key=YOUR-API-KEY", map_center_lon=180)
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```

Mapbox

> Mapbox traces are deprecated and may be removed in a future version of Plotly.py.

How Layers Work in Mapbox Tile Maps

Mapbox tile maps are composed of various layers, of three different types:

1. `layout.mapbox.style` defines is the lowest layers, also known as your "base map"
2. The various traces in `data` are by default rendered above the base map (although this can be controlled via the `below` attribute).
3. `layout.mapbox.layers` is an array that defines more layers that are by default rendered above the traces in `data` (although this can also be controlled via the `below` attribute).

Mapbox Access Tokens and When You Need Them

The word "mapbox" in the trace names and `layout.mapbox` refers to the Mapbox GL JS open-source library, which is integrated into Plotly.py.

If your basemap in `layout.mapbox.style` uses data from the Mapbox *service*, then you will need to register for a free account at https://mapbox.com/ (https://mapbox.com/) and obtain a Mapbox Access token. This token should be provided in `layout.mapbox.access_token` (or, if using Plotly Express, via the px.set_mapbox_access_token() configuration function).

If you basemap in `layout.mapbox.style` uses maps from the Stadia Maps service (https://www.stadiamaps.com) (see below for details), you'll need to register for a Stadia Maps account and token.

Base Maps in `layout.mapbox.style`

The accepted values for `layout.mapbox.style` are one of:

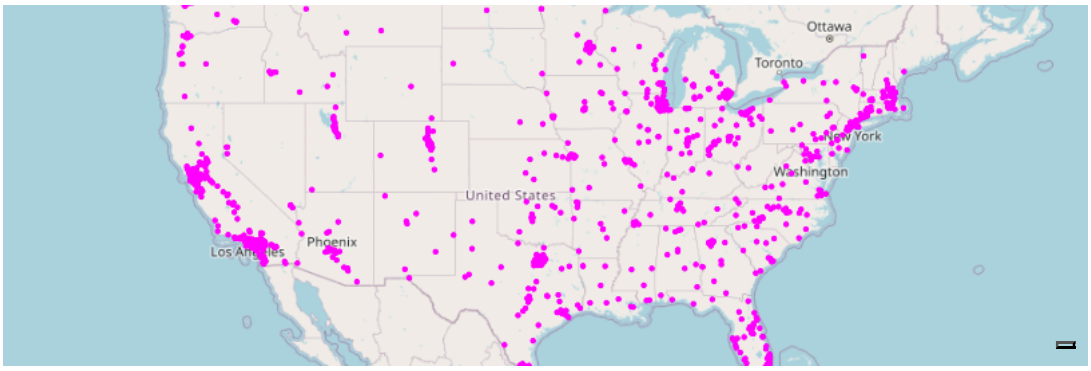
- `"white-bg"` yields an empty white canvas which results in no external HTTP requests
- `"open-street-map"`, `"carto-positron"`, and `"carto-darkmatter"` yield maps composed of *raster* tiles from various public tile servers which do not require signups or access tokens.
- `"basic"`, `"streets"`, `"outdoors"`, `"light"`, `"dark"`, `"satellite"`, or `"satellite-streets"` yield maps composed of *vector* tiles from the Mapbox service, and *do* require a Mapbox Access Token or an on-premise Mapbox installation.
- `"stamen-terrain"`, `"stamen-toner"` or `"stamen-watercolor"` yield maps composed of *raster* tiles from the Stadia Maps service (https://www.stadiamaps.com), and require a Stadia Maps account and token.
- A Mapbox service style URL, which requires a Mapbox Access Token or an on-premise Mapbox installation.
- A Mapbox Style object as defined at https://docs.mapbox.com/mapbox-gl-js/style-spec/ (https://docs.mapbox.com/mapbox-gl-js/style-spec/)

OpenStreetMap tiles: no token needed

Here's a simple map rendered with OpenStreetMaps tiles, without needing a Mapbox Access Token:

```
import pandas as pd
us_cities = pd.read_csv("https://raw.githubusercontent.com/plotly/datasets/master/us-cities-top-1k.csv")

import plotly.express as px

fig = px.scatter_mapbox(us_cities, lat="lat", lon="lon", hover_name="City", hover_data=["State", "Population"],
                        color_discrete_sequence=["fuchsia"], zoom=3, height=300)
fig.update_layout(mapbox_style="open-street-map")
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```

```
/tmp/ipykernel_17712/2481563717.py:6: DeprecationWarning:

*scatter_mapbox* is deprecated! Use *scatter_map* instead. Learn more at: https://plotly.com/python/mapbox-to-maplibre/
```



Using layout.mapbox.layers to Specify a Base Map

If you have access to your own private tile servers, or wish to use a tile server not included in the list above, the recommended approach is to set layout.mapbox.style to "white-bg" and to use layout.mapbox.layers with below to specify a custom base map.

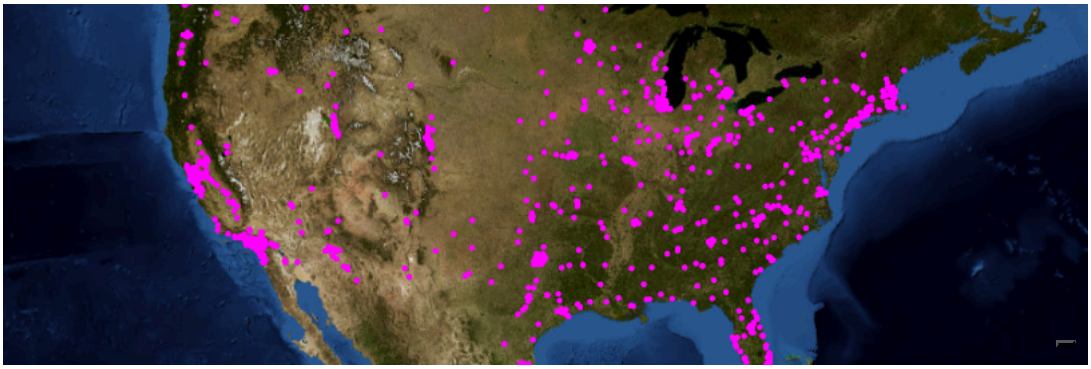
> If you omit the below attribute when using this approach, your data will likely be hidden by fully-opaque raster tiles!

Base Tiles from the USGS: no token needed

Here is an example of a map which uses a public USGS imagery map, specified in layout.mapbox.layers, and which is rendered *below* the data layer.

```
import pandas as pd
us_cities = pd.read_csv("https://raw.githubusercontent.com/plotly/datasets/master/us-cities-top-1k.csv")

import plotly.express as px

fig = px.scatter_mapbox(us_cities, lat="lat", lon="lon", hover_name="City", hover_data=["State", "Population"],
                        color_discrete_sequence=["fuchsia"], zoom=3, height=300)
fig.update_layout(
    mapbox_style="white-bg",
    mapbox_layers=[
        {
            "below": 'traces',
            "sourcetype": "raster",
            "sourceattribution": "United States Geological Survey",
            "source": [
                "https://basemap.nationalmap.gov/arcgis/rest/services/USGSImageryOnly/MapServer/tile/{z}/{y}/{x}"
            ]
        }
      ])
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```

```
/tmp/ipykernel_17712/1777570246.py:6: DeprecationWarning:

*scatter_mapbox* is deprecated! Use *scatter_map* instead. Learn more at: https://plotly.com/python/mapbox-to-maplibre/
```

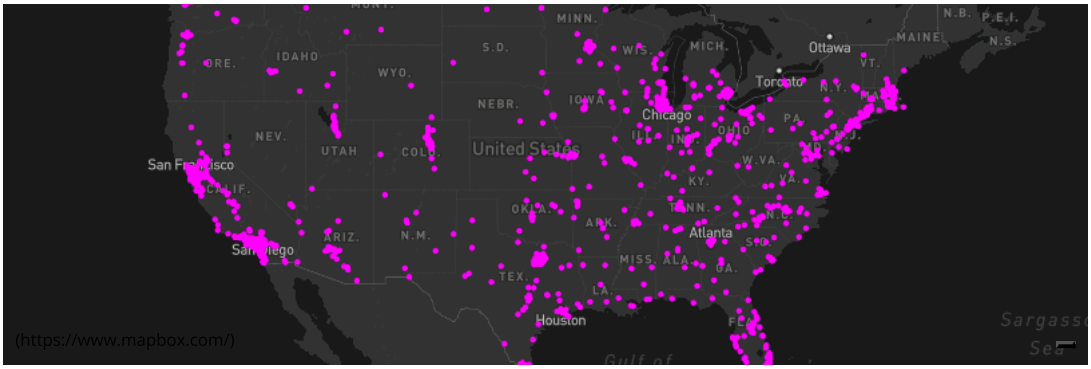


Dark tiles from Mapbox service: free token needed

Here is a map rendered with the "dark" style from the Mapbox service, which requires an Access Token:

```
token = open(".mapbox_token").read() # you will need your own token

import pandas as pd
us_cities = pd.read_csv("https://raw.githubusercontent.com/plotly/datasets/master/us-cities-top-1k.csv")

import plotly.express as px

fig = px.scatter_mapbox(us_cities, lat="lat", lon="lon", hover_name="City", hover_data=["State", "Population"],
                        color_discrete_sequence=["fuchsia"], zoom=3, height=300)
fig.update_layout(mapbox_style="dark", mapbox_accesstoken=token)
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```

```
/tmp/ipykernel_17712/373653669.py:8: DeprecationWarning:

*scatter_mapbox* is deprecated! Use *scatter_map* instead. Learn more at: https://plotly.com/python/mapbox-to-maplibre/
```

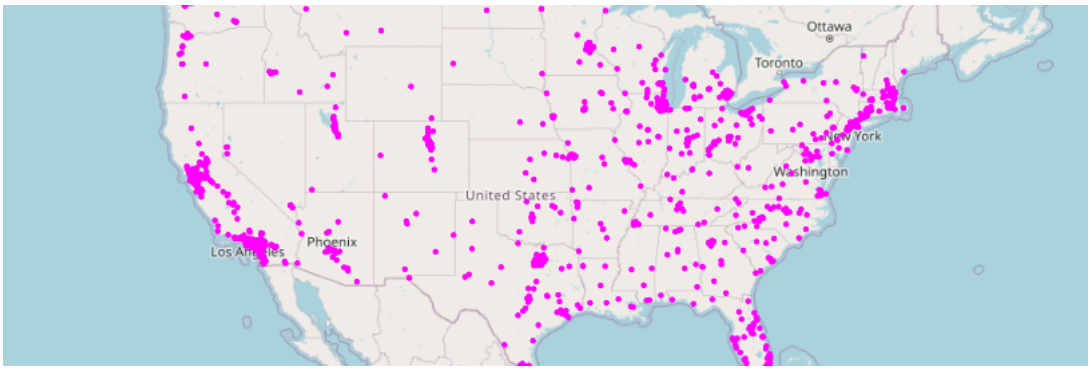


Setting Map Bounds

New in 5.11

Set bounds for a map to specify an area outside which a user interacting with the map can't pan or zoom. Here we set a maximum longitude of -180, a minimum longitude of -50, a maximum latitude of 90, and a minimum latitude of 20.

```
import plotly.express as px
import pandas as pd

us_cities = pd.read_csv(
    "https://raw.githubusercontent.com/plotly/datasets/master/us-cities-top-1k.csv"
)

fig = px.scatter_map(
    us_cities,
    lat="lat",
    lon="lon",
    hover_name="City",
    hover_data=["State", "Population"],
    color_discrete_sequence=["fuchsia"],
    zoom=3,
    height=300,
)
fig.update_layout(map_style="open-street-map")
fig.update_layout(margin={"r": 0, "t": 0, "l": 0, "b": 0})
fig.update_layout(map_bounds={"west": -180, "east": -50, "south": 20, "north": 90})
fig.show()
```



Reference

See https://plotly.com/python/reference/layout/map/ (https://plotly.com/python/reference/layout/map/) for more information and options on Maplibre-based tile maps and https://plotly.com/python/reference/layout/mapbox/ (https://plotly.com/python/reference/layout/mapbox/) for Mapbox-based tile maps.

What About Dash?

Dash (https://dash.plot.ly/) is an open-source framework for building analytical applications, with no Javascript required, and it is tightly integrated with the Plotly graphing library.

Learn about how to install Dash at https://dash.plot.ly/installation (https://dash.plot.ly/installation).

Everywhere in this page that you see fig.show(), you can display the same figure in a Dash application by passing it to the figure argument of the Graph component (https://dash.plot.ly/dash-core-components/graph) from the built-in dash_core_components package like this:

```
import plotly.graph_objects as go # or plotly.express as px
fig = go.Figure() # or any Plotly Express function e.g. px.bar(...)
# fig.add_trace( ... )
# fig.update_layout( ... )

from dash import Dash, dcc, html

app = Dash()
app.layout = html.Div([
    dcc.Graph(figure=fig)

use_reloader=False)  # Turn off reloader if inside Jupyter
```