st.map

Streamlit Version Version 1.41.0

Display a map with a scatterplot overlaid onto it.

This is a wrapper around `st.pydeck_chart` to quickly create scatterplot charts on top of a map, with auto-centering and auto-zoom.

When using this command, Mapbox provides the map tiles to render map content. Note that Mapbox is a third-party product and Streamlit accepts no responsibility or liability of any kind for Mapbox or for any content or information made available by Mapbox.

Mapbox requires users to register and provide a token before users can request map tiles. Currently, Streamlit provides this token for you, but this could change at any time. We strongly recommend all users create and use their own personal Mapbox token to avoid any disruptions to their experience. You can do this with the `mapbox.token` config option. The use of Mapbox is governed by Mapbox's Terms of Use.

To get a token for yourself, create an account at https://mapbox.com. For more info on how to set config options, see https://docs.streamlit.io/develop/api-reference/configuration/config.toml.

Function signature[source]

st.map(data=None, *, latitude=None, longitude=None, color=None, size=None, zoom=None, use_container_width=True, width=None, height=None)

Parameters	
data (Anything supported by st.dataframe)	The data to be plotted.
latitude (str or None)	The name of the column containing the latitude coordinates of the datapoints in the chart. If None, the latitude data will come from any column named 'lat', 'latitude', 'LAT', or 'LATITUDE'.
longitude (str or None)	The name of the column containing the longitude coordinates of the datapoints in the chart. If None, the longitude data will come from any column named 'lon', 'longitude', 'LON', or 'LONGITUDE'.
color (str or tuple or None)	The color of the circles representing each datapoint. Can be: • None, to use the default color. • A hex string like "#ffaa00" or "#ffaa0088". • An RGB or RGBA tuple with the red, green, blue, and alpha components specified as ints from 0 to 255 or floats from 0.0 to 1.0. • The name of the column to use for the color. Cells in this column should contain colors represented as a hex string or color tuple, as described above.
size (str or float or None)	The size of the circles representing each point, in meters. This can be: • None, to use the default size. • A number like 100, to specify a single size to use for all datapoints.

Function signature[source]

st.map(data=None, *, latitude=None, longitude=None, color=None, size=None, zoom=None, use_container_width=True, width=None, height=None)

- The name of the column to use for the size. This allows each datapoint to be represented by a circle of a different size.

zoom (int)	Zoom level as specified in https://wiki.openstreetmap.org/wiki/Zoom_levels.
use_container_width (bool)	Whether to override the map's native width with the width of the parent container. If `use_container_width` is `True` (default), Streamlit sets the width of the map to match the width of the parent container. If `use_container_width` is `False`, Streamlit sets the width of the chart to fit its contents according to the plotting library, up to the width of the parent container.
width (int or None)	Desired width of the chart expressed in pixels. If `width` is `None` (default), Streamlit sets the width of the chart to fit its contents according to the plotting library, up to the width of the parent container. If `width` is greater than the width of the parent container, Streamlit sets the chart width to match the width of the parent container. To use `width`, you must set `use_container_width=False`.
height (int or None)	Desired height of the chart expressed in pixels. If `height` is `None` (default), Streamlit sets the height of the chart to fit its contents according to the plotting library.

Examples

```
import streamlit as st
import pandas as pd
import numpy as np

df = pd.DataFrame(
    np.random.randn(1000, 2) / [50, 50] + [37.76, -122.4],
    columns=["lat", "lon"],
)
st.map(df)
```

Built with Streamlit 🎈
Fullscreen *open_in_new*

You can also customize the size and color of the datapoints:

```
st.map(df, size=20, color="#0044ff")
```

And finally, you can choose different columns to use for the latitude and longitude components, as well as set size and color of each datapoint dynamically based on other columns:

```
import streamlit as st
import pandas as pd
import numpy as np

df = pd.DataFrame(
    {
        "col1": np.random.randn(1000) / 50 + 37.76,
        "col2": np.random.randn(1000) / 50 + -122.4,
        "col3": np.random.randn(1000) * 100,
        "col4": np.random.rand(1000, 4).tolist(),
    }
)

st.map(df, latitude="col1", longitude="col2", size="col3", color="col4")
```

Built with Streamlit 🎈
Fullscreen *open_in_new*

element.add_rows

Streamlit Version Version 1.41.0

Concatenate a dataframe to the bottom of the current one.

Function signature[source]

element.add_rows(data=None, **kwargs)

Parameters

data (pandas.DataFrame, pandas.Styler, pyarrow.Table, numpy.ndarray, pyspark.sql.DataFrame, snowflake.snowpark.dataframe.DataFrame, Iterable, dict, or None)	Table to concat. Optional.

Function signature[source]

element.add_rows(data=None, **kwargs)

**kwargs (pandas.DataFrame, numpy.ndarray, Iterable, dict, or None)	The named dataset to concat. Optional. You can only pass in 1 dataset (including the one in the data parameter).

Example

```
import streamlit as st
import pandas as pd
import numpy as np

df1 = pd.DataFrame(
    np.random.randn(50, 20), columns=("col %d" % i for i in range(20))
)

my_table = st.table(df1)

df2 = pd.DataFrame(
    np.random.randn(50, 20), columns=("col %d" % i for i in range(20))
)

my_table.add_rows(df2)
# Now the table shown in the Streamlit app contains the data for
# df1 followed by the data for df2.
```

You can do the same thing with plots. For example, if you want to add more data to a line chart:

```
# Assuming df1 and df2 from the example above still exist...
my_chart = st.line_chart(df1)
my_chart.add_rows(df2)
# Now the chart shown in the Streamlit app contains the data for
# df1 followed by the data for df2.
```

And for plots whose datasets are named, you can pass the data with a keyword argument where the key is the name:

```
my_chart = st.vega_lite_chart(
    {
        "mark": "line",
        "encoding": {"x": "a", "y": "b"},
        "datasets": {
            "some_fancy_name": df1,  # <-- named dataset
        },
        "data": {"name": "some_fancy_name"},
    }
)
my_chart.add_rows(some_fancy_name=df2)  # <-- name used as keyword
```

←Previous: st.line_chartNext: st.scatter_chart→

forum

Still have questions?

Our forums are full of helpful information and Streamlit experts.