st.pydeck_chart

Streamlit Version Version 1.41.0

Draw a chart using the PyDeck library.

This supports 3D maps, point clouds, and more! More info about PyDeck at https://deckgl.readthedocs.io/en/latest/.

These docs are also quite useful:

- DeckGL docs: https://github.com/uber/deck.gl/tree/master/docs
- DeckGL JSON docs: https://github.com/uber/deck.gl/tree/master/modules/json

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.pydeck_chart(pydeck_obj=None, *, use_container_width=False, width=None, height=None, selection_mode="single-object", on_select="ignore", key=None)

Parameters	
pydeck_obj (pydeck.Deck or None)	Object specifying the PyDeck chart to draw.
use_container_width (bool)	Whether to override the figure's native width with the width of the parent container. If `use_container_width` is `False` (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 `use_container_width` is `True`, Streamlit sets the width of the figure to match 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.
Returns	
(element or dict)	If `on_select` is `"ignore"` (default), this command returns an internal placeholder for the chart element. Otherwise, this method returns a dictionary-like object that supports both key and attribute notation. The attributes are described by the `PydeckState` dictionary schema.

Function signature[source]

st.pydeck_chart(pydeck_obj=None, *, use_container_width=False, width=None, height=None, selection_mode="single-object", on_select="ignore", key=None)

on_select ("ignore" or "rerun" or callable)	How the figure should respond to user selection events. This controls whether or not the chart behaves like an input widget. `on_select` can be one of the following: • `"ignore"` (default): Streamlit will not react to any selection events in the chart. The figure will not behave like an input widget. • `"rerun"`: Streamlit will rerun the app when the user selects data in the chart. In this case, `st.pydeck_chart` will return the selection data as a dictionary. • A `callable`: Streamlit will rerun the app and execute the callable as a callback function before the rest of the app. In this case, `st.pydeck_chart` will return the selection data as a dictionary. If `on_select` is not `"ignore"`, all layers must have a declared `id` to keep the chart stateful across reruns.
selection_mode ("single-object" or "multi-object")	The selection mode of the chart. This can be one of the following: • `"single-object"` (default): Only one object can be selected at a time. • `"multi-object"`: Multiple objects can be selected at a time.
key (str)	An optional string to use for giving this element a stable identity. If `key` is `None` (default), this element's identity will be determined based on the values of the other parameters. Additionally, if selections are activated and `key` is provided, Streamlit will register the key in Session State to store the selection state. The selection state is read-only.
Returns	
(element or dict)	If `on_select` is `"ignore"` (default), this command returns an internal placeholder for the chart element. Otherwise, this method returns a dictionary-like object that supports both key and attribute notation. The attributes are described by the `PydeckState` dictionary schema.

Example

Here's a chart using a HexagonLayer and a ScatterplotLayer. It uses either the light or dark map style, based on which Streamlit theme is currently active:

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

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

st.pydeck_chart(
    pdk.Deck(
        map_style=None,
        initial_view_state=pdk.ViewState(
            latitude=37.76,
            longitude=-122.4,
```

```
            zoom=11,
            pitch=50,
        ),
        layers=[
            pdk.Layer(
                "HexagonLayer",
                data=chart_data,
                get_position="[lon, lat]",
                radius=200,
                elevation_scale=4,
                elevation_range=[0, 1000],
                pickable=True,
                extruded=True,
            ),
            pdk.Layer(
                "ScatterplotLayer",
                data=chart_data,
                get_position="[lon, lat]",
                get_color="[200, 30, 0, 160]",
                get_radius=200,
            ),
        ],
    )
)
```

Built with Streamlit 🎈
Fullscreen *open_in_new*

Note

To make the PyDeck chart's style consistent with Streamlit's theme, you can set `map_style=None` in the `pydeck.Deck` object.

Chart selections

PydeckState

Streamlit Version Version 1.41.0

The schema for the PyDeck event state.

The event state is stored in a dictionary-like object that supports both key and attribute notation. Event states cannot be programmatically changed or set through Session State.

Only selection events are supported at this time.

Attributes

selection (dict)	The state of the `on_select` event. This attribute returns a dictionary-like object that supports both key and attribute notation. The attributes are described by the `PydeckSelectionState` dictionary schema.

PydeckSelectionState

Streamlit Version Version 1.41.0

The schema for the PyDeck chart selection state.

The selection state is stored in a dictionary-like object that supports both key and attribute notation. Selection states cannot be programmatically changed or set through Session State.

You must define `id` in `pydeck.Layer` to ensure statefulness when using selections with `st.pydeck_chart`.

Attributes

indices (dict[str, list[int]])	A dictionary of selected objects by layer. Each key in the dictionary is a layer id, and each value is a list of object indices within that layer.
objects (dict[str, list[dict[str, Any]]])	A dictionary of object attributes by layer. Each key in the dictionary is a layer id, and each value is a list of metadata dictionaries for the selected objects in that layer.

Examples

The following example has multi-object selection enabled. The chart displays US state capitals by population (2023 US Census estimate). You can access this data from GitHub.

```
import streamlit as st
import pydeck
import pandas as pd

capitals = pd.read_csv(
    "capitals.csv",
    header=0,
    names=[
        "Capital",
```

```
        "State",
        "Abbreviation",
        "Latitude",
        "Longitude",
        "Population",
    ],
)
capitals["size"] = capitals.Population / 10

point_layer = pydeck.Layer(
    "ScatterplotLayer",
    data=capitals,
    id="capital-cities",
    get_position=["Longitude", "Latitude"],
    get_color="[255, 75, 75]",
    pickable=True,
    auto_highlight=True,
    get_radius="size",
)

view_state = pydeck.ViewState(
    latitude=40, longitude=-117, controller=True, zoom=2.4, pitch=30
)

chart = pydeck.Deck(
    point_layer,
    initial_view_state=view_state,
    tooltip={"text": "{Capital}, {Abbreviation}\nPopulation: {Population}"},
)

event = st.pydeck_chart(chart, on_select="rerun", selection_mode="multi-object")

event.selection
```

Built with Streamlit 🎈
Fullscreen *open_in_new*

This is an example of the selection state when selecting a single object from a layer with id, `"captial-cities"`:


```
{
  "indices":{
    "capital-cities":[
      2
    ]
  },
  "objects":{
    "capital-cities":[
      {
        "Abbreviation":" AZ"
        "Capital":"Phoenix"
        "Latitude":33.448457
        "Longitude":-112.073844
        "Population":1650070
        "State":" Arizona"
        "size":165007.0
```

```
        }
      ]
    }
}
```

←Previous: st.plotly_chartNext: st.pyplot→

forum

Still have questions?

Our forums are full of helpful information and Streamlit experts.

© 2025 Snowflake Inc. Cookie policy

forum Ask AI