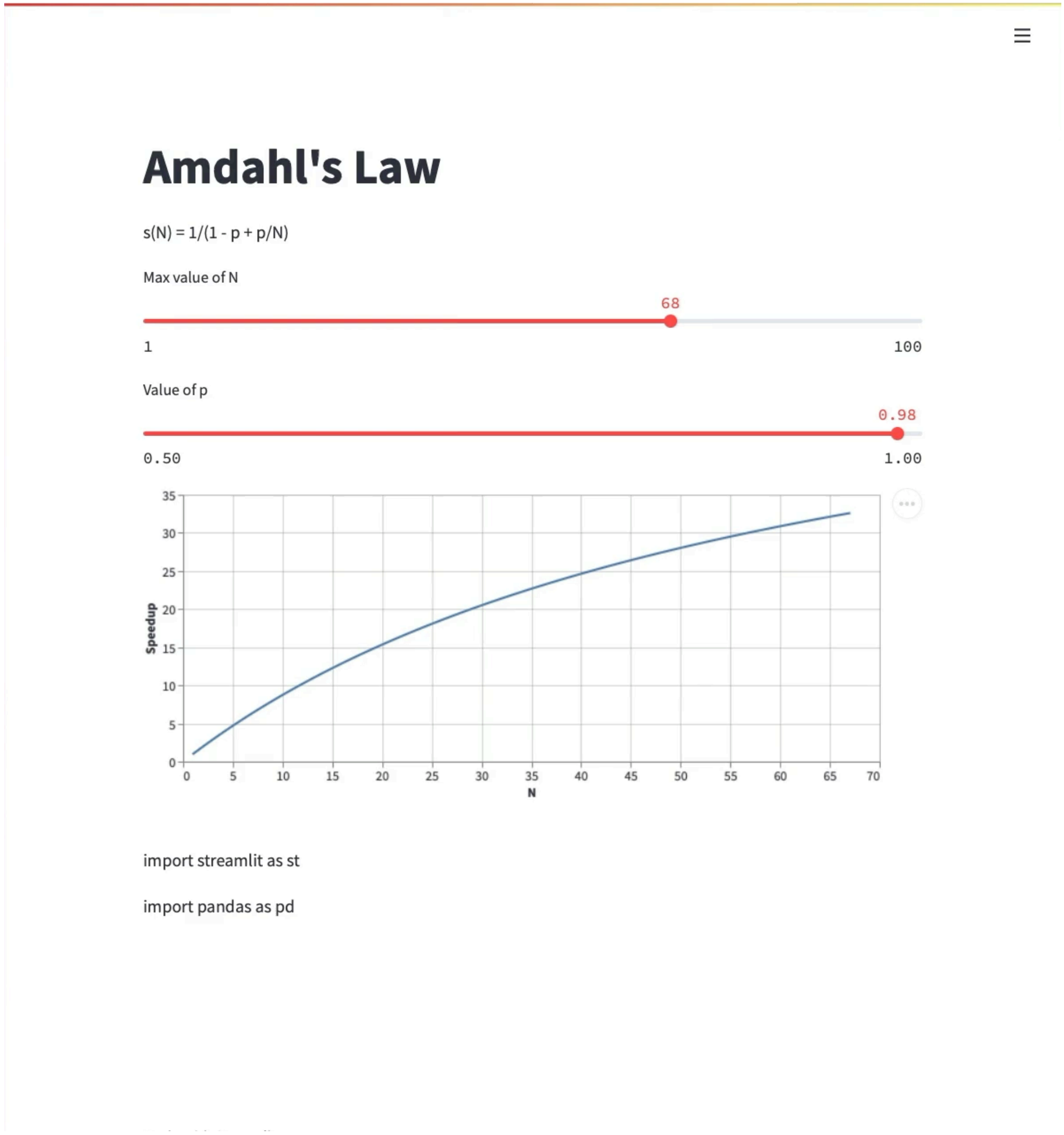
CS 649 Big Data: Tools and Methods
Spring Semester, 2022
Doc 9 Dashboards
Feb 8, 2022

Dashboards & Data Web Apps



Amdahl's Law

s(N) = 1/(1 - p + p/N)

Max value of N

68

1 100

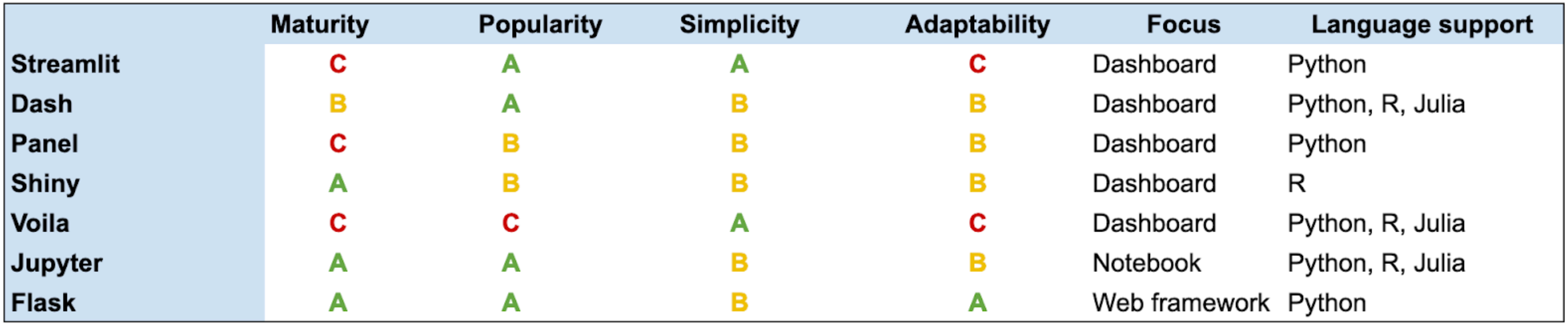
Value of p

0.98

0.50 1.00

```
import streamlit as st

import pandas as pd
```

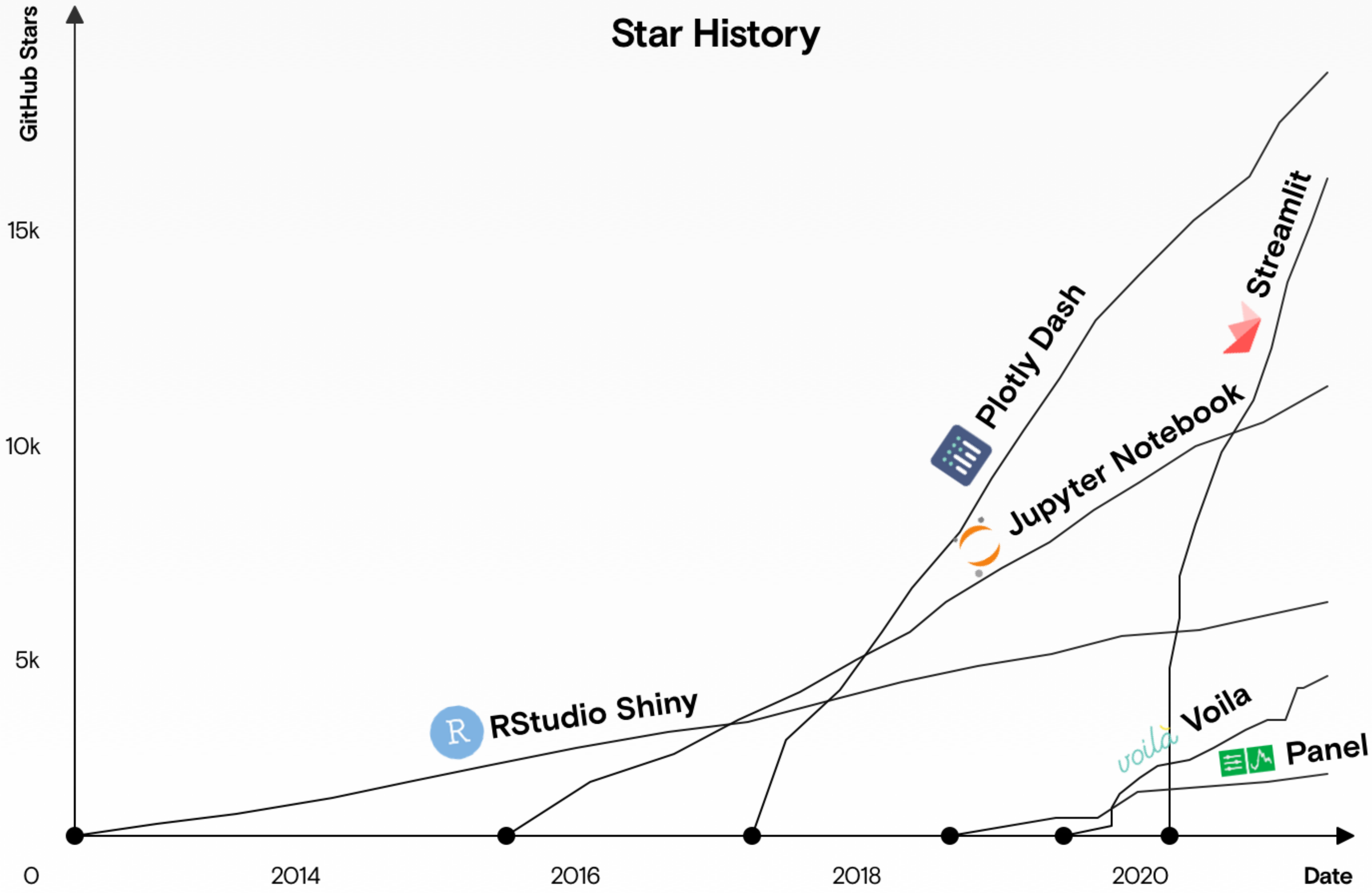
Dashboards



	Maturity	Popularity	Simplicity	Adaptability	Focus	Language support
Streamlit	C	A	A	C	Dashboard	Python
Dash	B	A	B	B	Dashboard	Python, R, Julia
Panel	C	B	B	B	Dashboard	Python
Shiny	A	B	B	B	Dashboard	R
Voila	C	C	A	C	Dashboard	Python, R, Julia
Jupyter	A	A	B	B	Notebook	Python, R, Julia
Flask	A	A	B	A	Web framework	Python

A good overview at:

https://www.datarevenue.com/en-blog/data-dashboarding-streamlit-vs-dash-vs-shiny-vs-voila

Star History
GitHub Stars
15k
10k
5k
0
2014
2016
2018
2020
Date
RStudio Shiny
Plotly Dash
Jupyter Notebook
Streamlit
voila
Voila
Panel

Streamlit Demo Source

```
import streamlit as st
import pandas as pd

st.title("Amdahl's Law")
st.write("s(N) = 1/(1 - p + p/N)")

def speed_up(N,p):
   return 1/(1 - p + p/N)

max_N = st.slider("Max value of N", min_value=1, max_value=100, step=1, value=5)
p = st.slider("Value of p", min_value=.5, max_value=1.0, step=0.001, value=0.8)

N_series = pd.Series(range(1,max_N), index=range(1,max_N))

st.line_chart(speed_up(N_series, p))
```

Streamlit

https://streamlit.io

Simple to create web apps to display, visualize data

Host on
- AWS
- Heroku
- Azure
- Google Cloud Platform
- Digital Ocean

App
- Windows
- MacOS
- Android

Installing & Developing

https://docs.streamlit.io/library/get-started/installation

Create python program in .py file:

Say example.py

Run using:

streamline run example.py

Opens web app in browser

Changes to example.py are reflected in the browser

Data Flow

When screen needs to updated

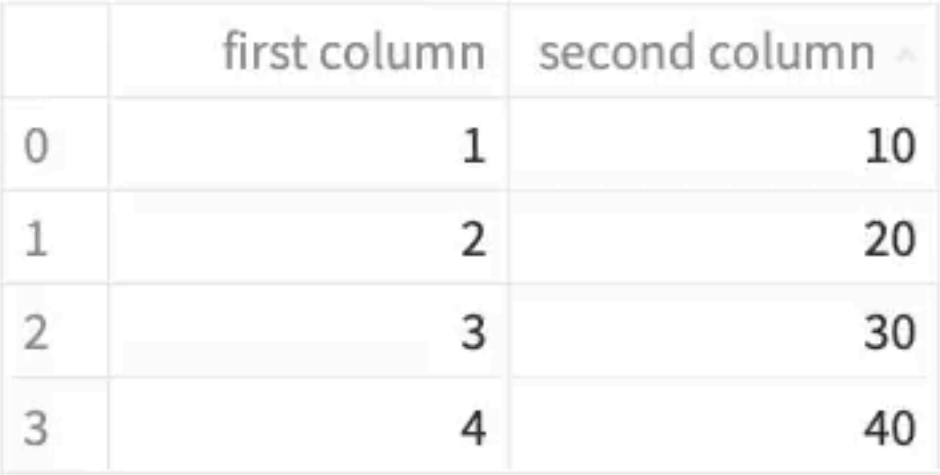
Entire program is rerun

Can cache computations to avoid rerunning them

```
import streamlit as st

@st.cache  # 👈 This function will be cached
def my_slow_function(arg1, arg2):
    # Do something really slow in here!
    return the_output
```

Displaying Data - magic

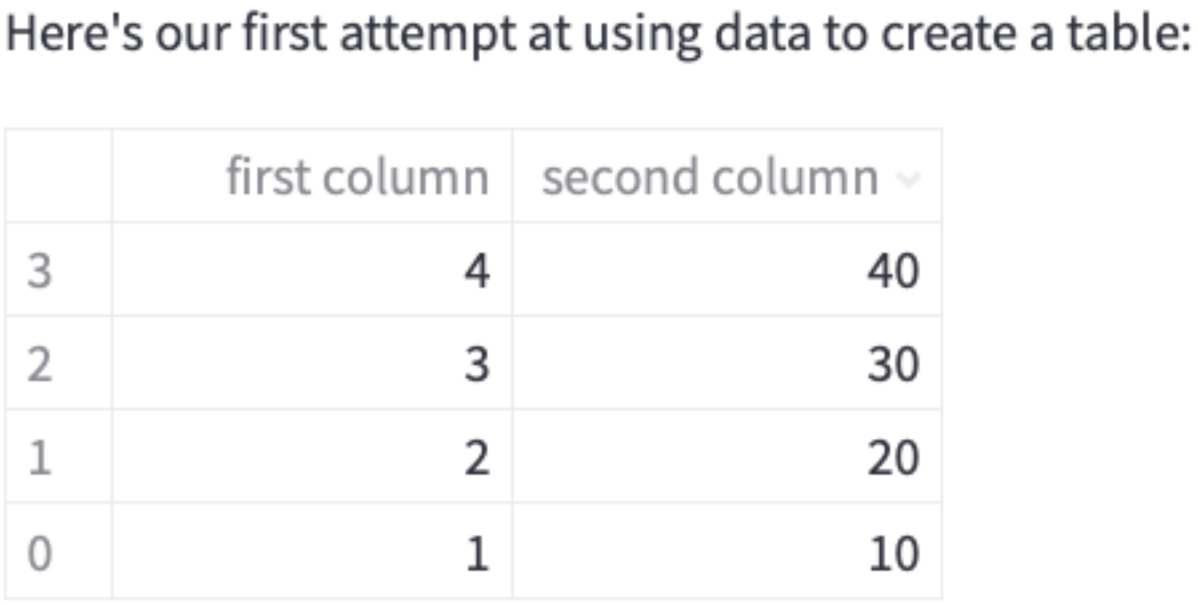
```
import streamlit as st
import pandas as pd
df = pd.DataFrame({
  'first column': [1, 2, 3, 4],
  'second column': [10, 20, 30, 40]
})

df
```



	first column	second column
0	1	10
1	2	20
2	3	30
3	4	40

Displaying Data - st.write

```
import streamlit as st
import pandas as pd

st.write("Here's our first attempt at using data to create a table:")
st.write(pd.DataFrame({
    'first column': [1, 2, 3, 4],
    'second column': [10, 20, 30, 40]
}))
```

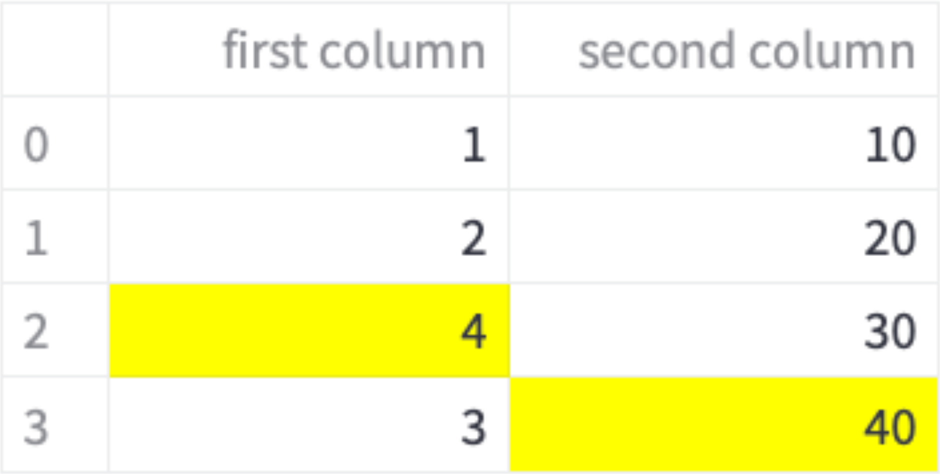


Here's our first attempt at using data to create a table:

	first column	second column
3	4	40
2	3	30
1	2	20
0	1	10

Highlight

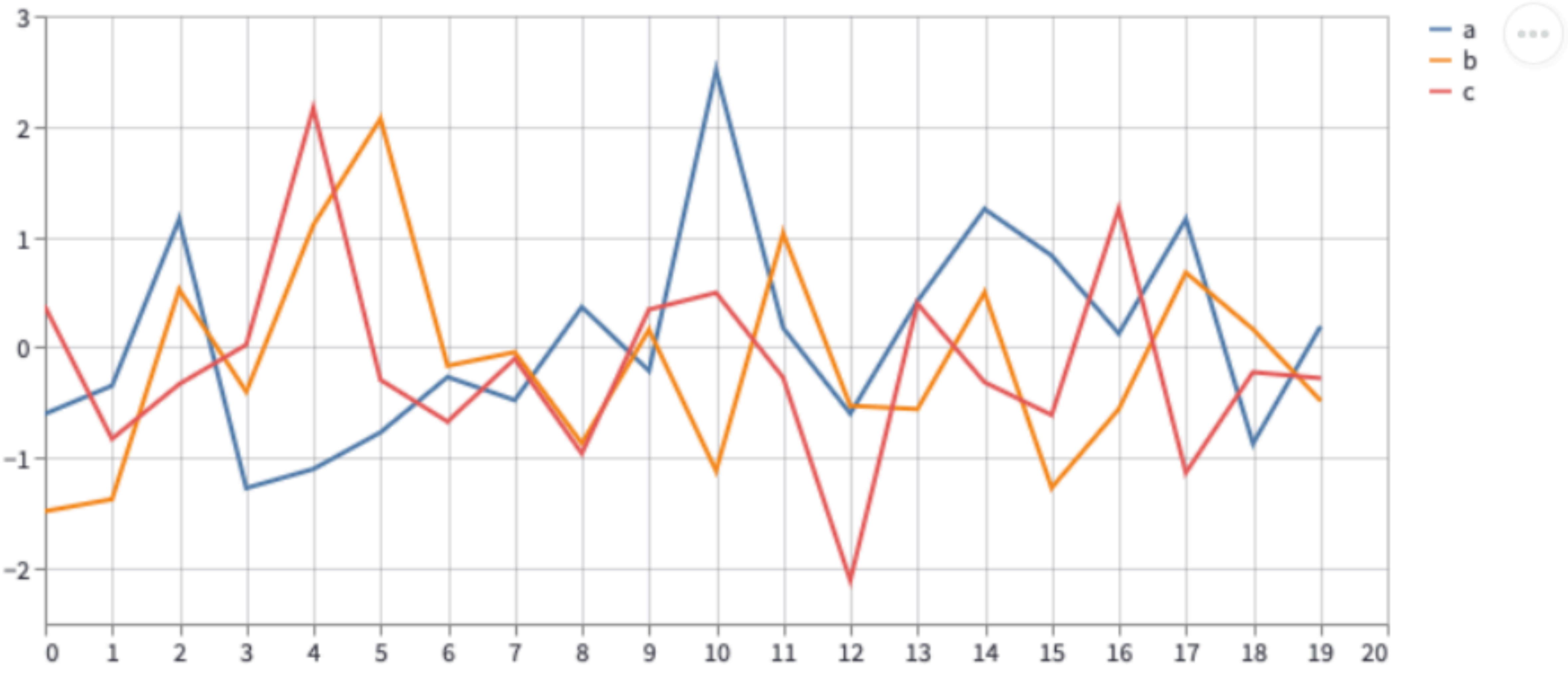
```
import streamlit as st
import pandas as pd

data = pd.DataFrame({
    'first column': [1, 2, 4, 3],
    'second column': [10, 20, 30, 40]
})

st.dataframe(data.style.highlight_max(axis=0))
```



	first column	second column
0	1	10
1	2	20
2	4	30
3	3	40

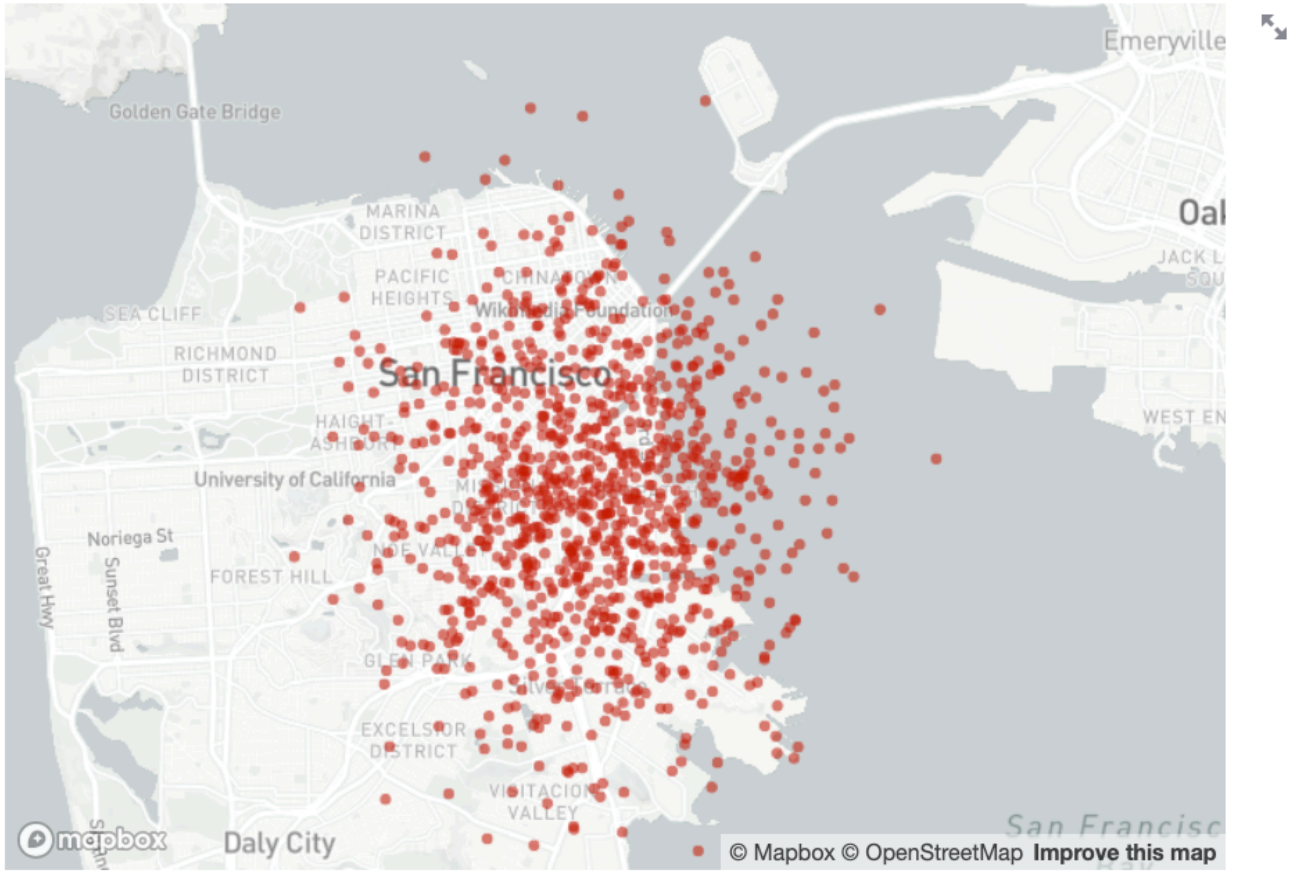
Line Chart

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

chart_data = pd.DataFrame(
     np.random.randn(20, 3),
     columns=['a', 'b', 'c'])

st.line_chart(chart_data)
```



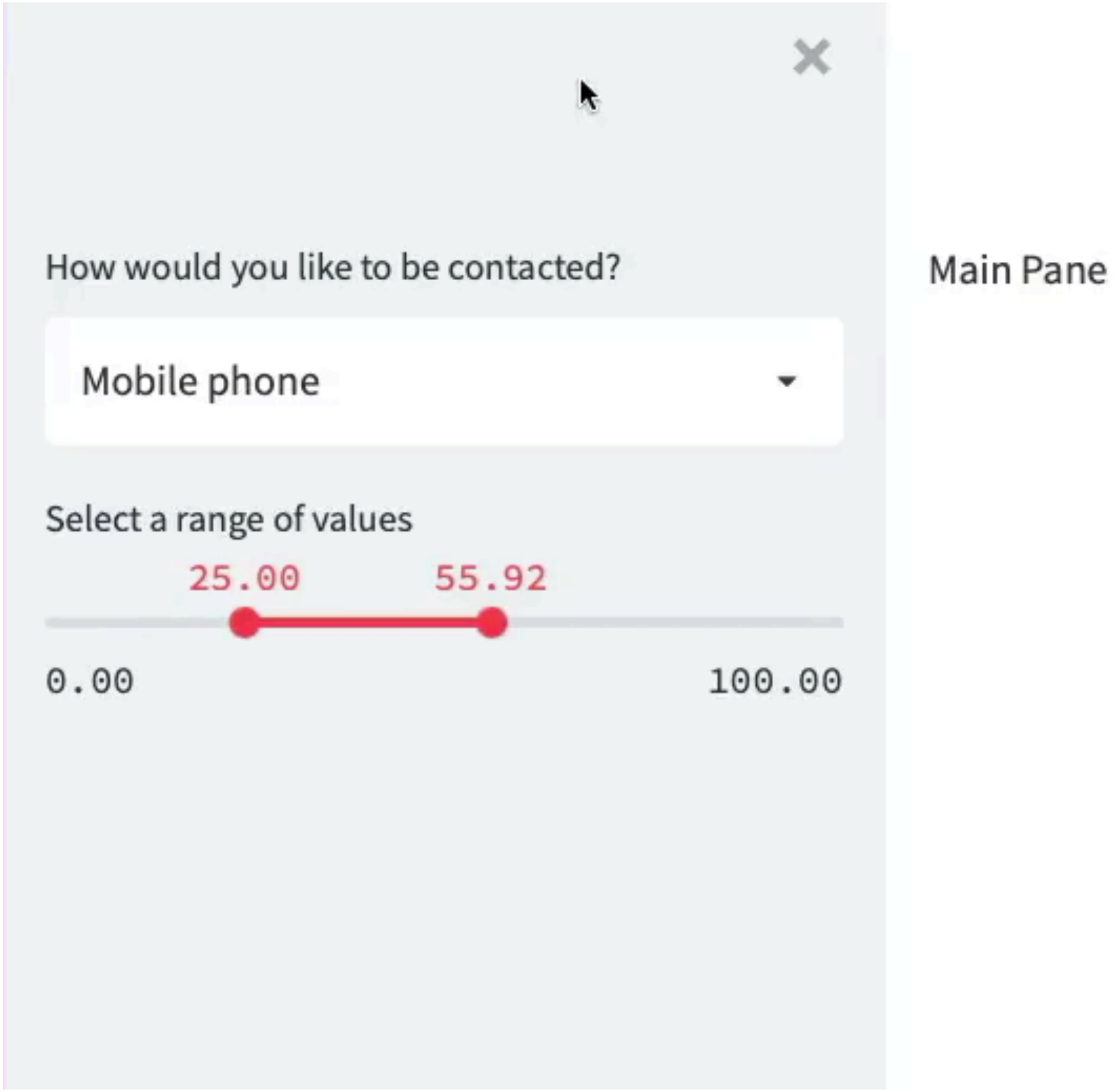
Maps

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

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

st.map(map_data)
```



Side Pane

```
import streamlit as st

# Add a selectbox to the sidebar:
add_selectbox = st.sidebar.selectbox(
    'How would you like to be contacted?',
    ('Email', 'Home phone', 'Mobile phone')
)

# Add a slider to the sidebar:
add_slider = st.sidebar.slider(
    'Select a range of values',
    0.0, 100.0, (25.0, 75.0)
)

st.write("Main Pane")
```



How would you like to be contacted?

Mobile phone

Select a range of values

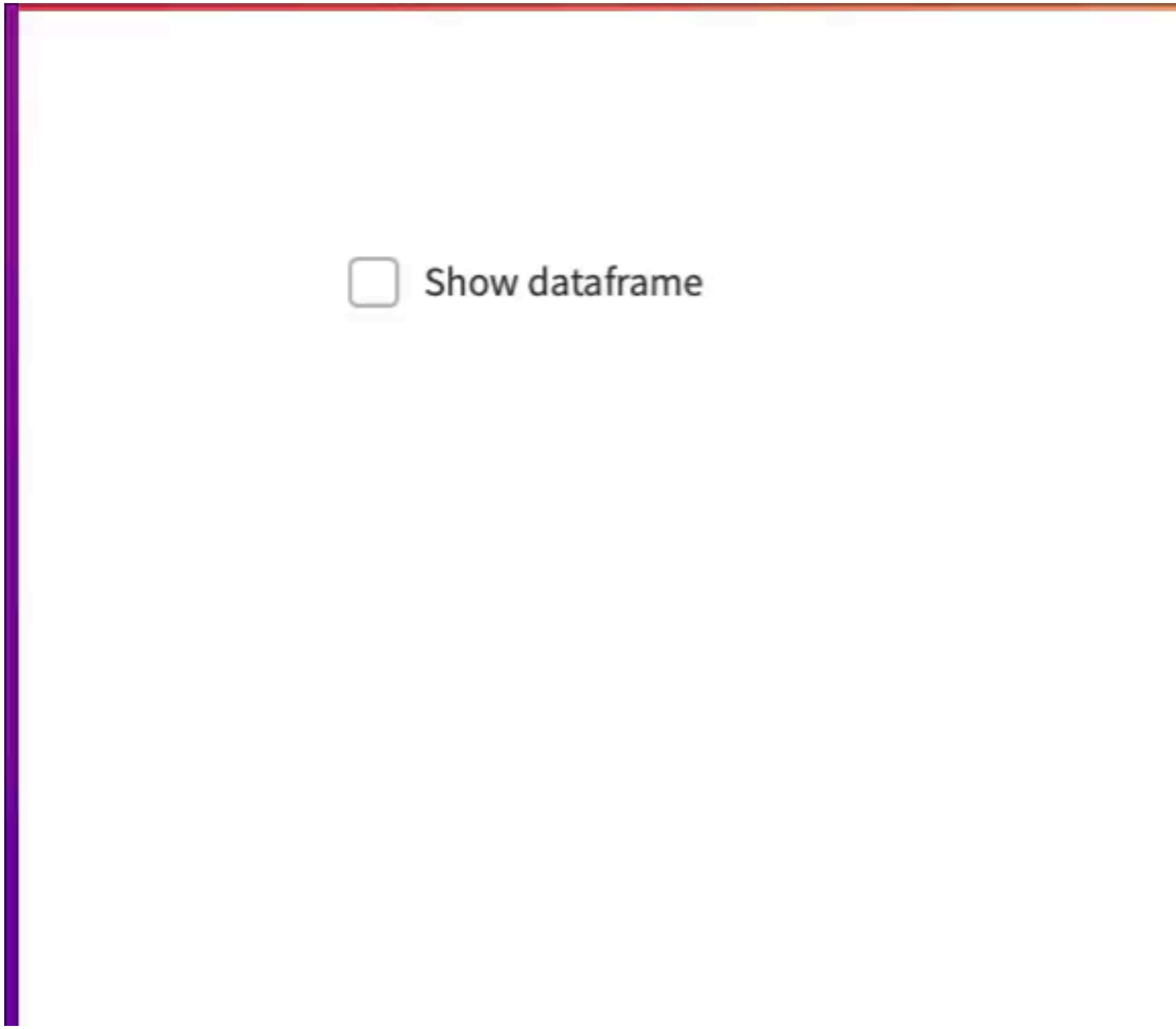
25.00 55.92

0.00 100.00

Main Pane

CheckBoxes

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

if st.checkbox('Show dataframe'):
    chart_data = pd.DataFrame(
       np.random.randn(20, 3),
       columns=['a', 'b', 'c'])

    chart_data
```



Some Widgets

st.line_chart

st.area_chart

st.bar_chart

st.pyplot

st.altair_chart

st.vega_lite_chart

st.plotly_chart

st.bokeh_chart

st.pydeck_chart

st.graphviz_chart

st.map

st.button

st.download_button

st.checkbox

st.radio

st.selectbox

st.multiselect

st.slider

st.select_slider

st.text_input

st.number_input

st.text_area

st.date_input

st.time_input

st.file_uploader

st.camera_input

st.color_picker

Workflow Issues

When screen needs to updated

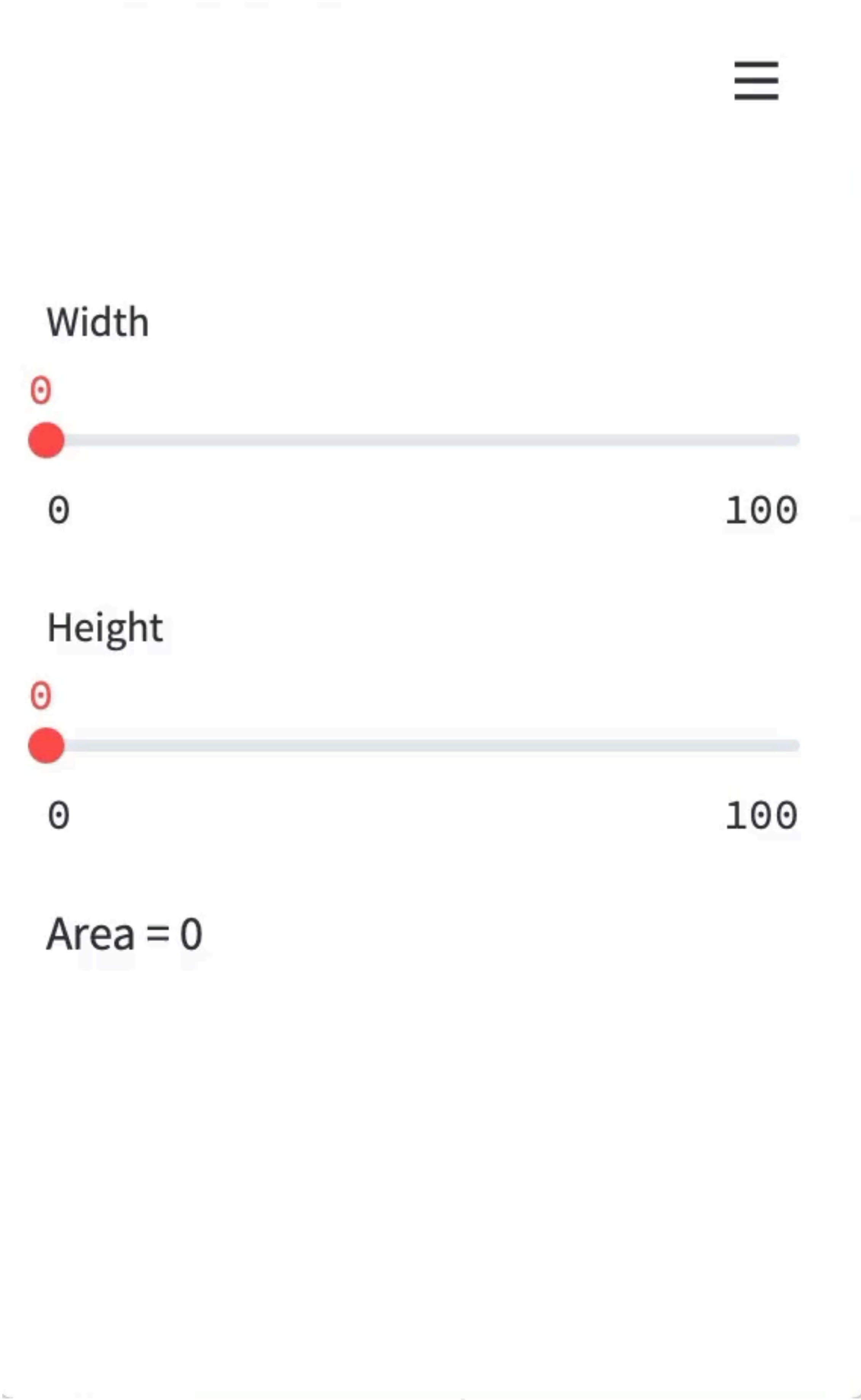
- Entire program is rerun

This causes runtime problems

- Long running functions get repeated
- When multiple widgets update the screen
- Session state - maintaining state

Example

```
import streamlit as st
import time

width = st.slider("Width")
height = st.slider("Height")

def compute():
    # Simulate a long running function
    with st.spinner('Working on it'):
        time.sleep(3)
    return width * height

st.write("Area = " + str(compute()))
```



Width

0

0 100

Height

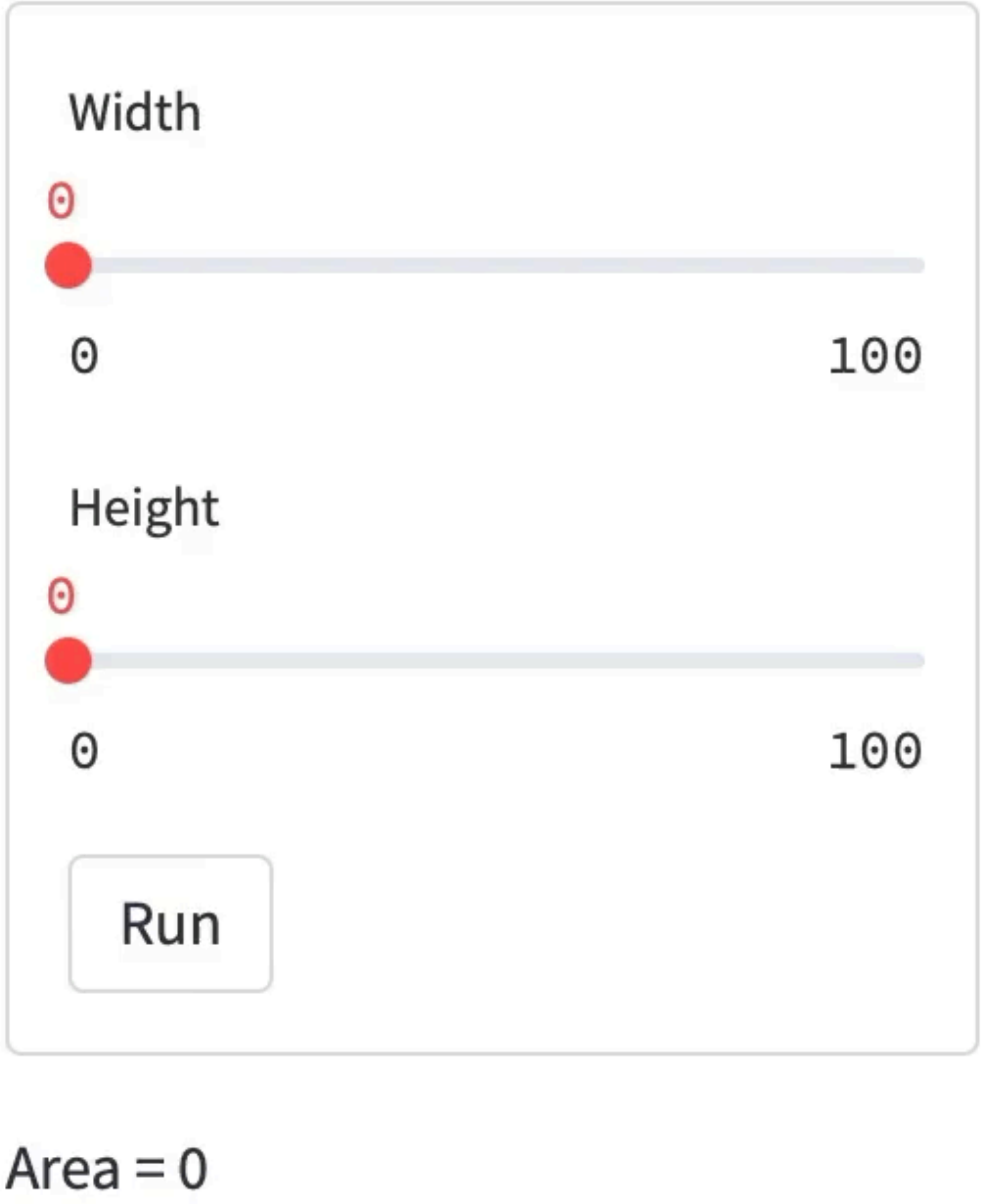
0

0 100

Area = 0

Using Forms

```
import streamlit as st
import time

with st.form("Compute_Values"):
    width = st.slider("Width")
    height = st.slider("Height")
    submitted = st.form_submit_button("Run")

def compute():
    # Simulate a long running function
    with st.spinner('Working on it'):
        time.sleep(3)
    return width * height

st.write("Area = " + str(compute()))
```

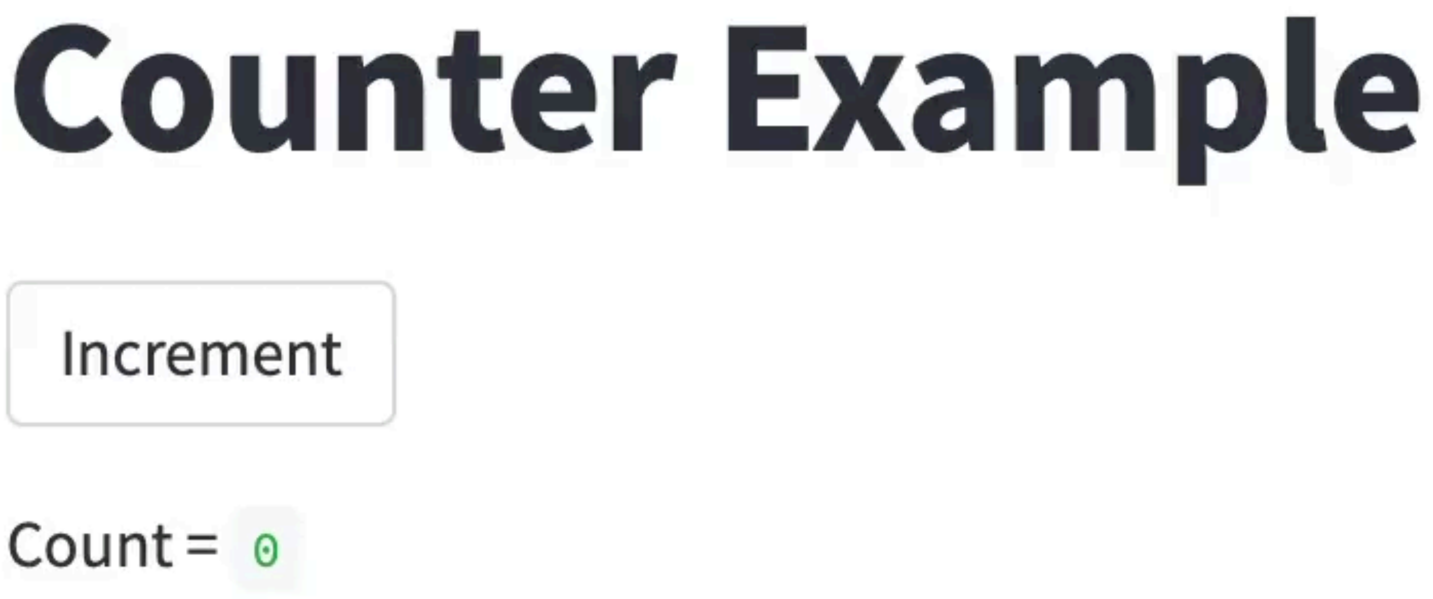


Area = 0

Maintaining State

```
import streamlit as st

st.title('Counter Example')
count = 0

increment = st.button('Increment')
if increment:
    count += 1

st.write('Count = ', count)
```

Count is set to 0 each time button is pressed

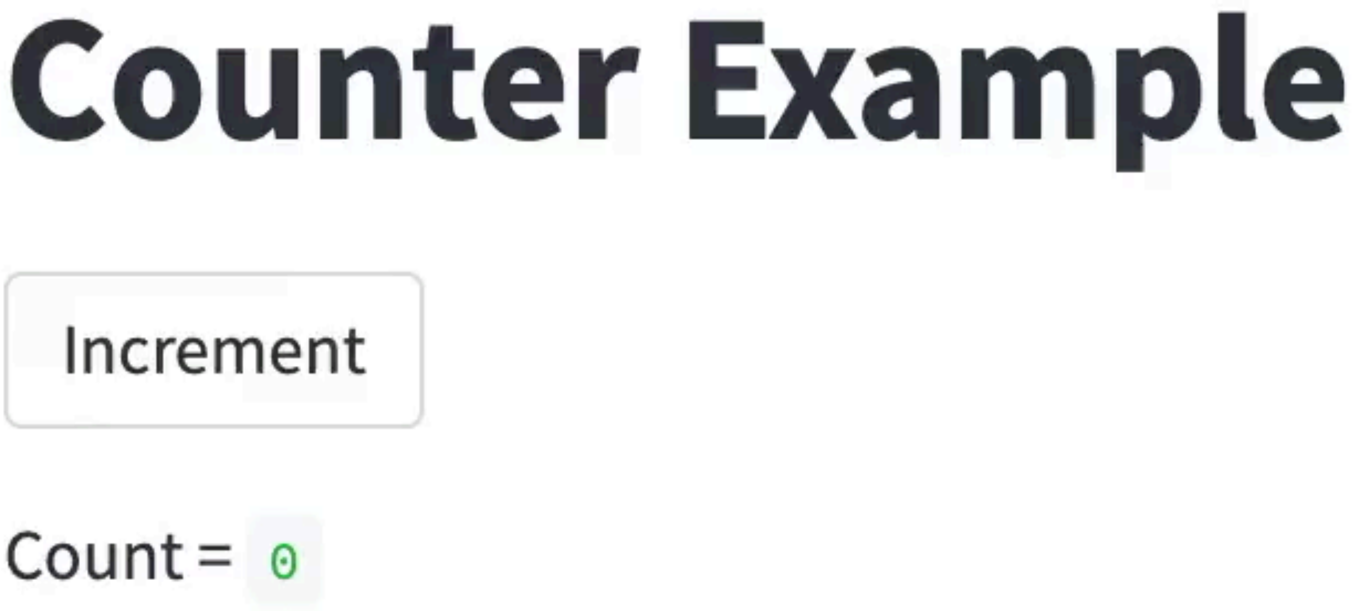


Counter Example

Count = 0

Using Session State

```
import streamlit as st

st.title('Counter Example')
if 'count' not in st.session_state:
    st.session_state.count = 0

increment = st.button('Increment')
if increment:
    st.session_state.count += 1

st.write('Count = ', st.session_state.count)
```



Counter Example

Count = 0

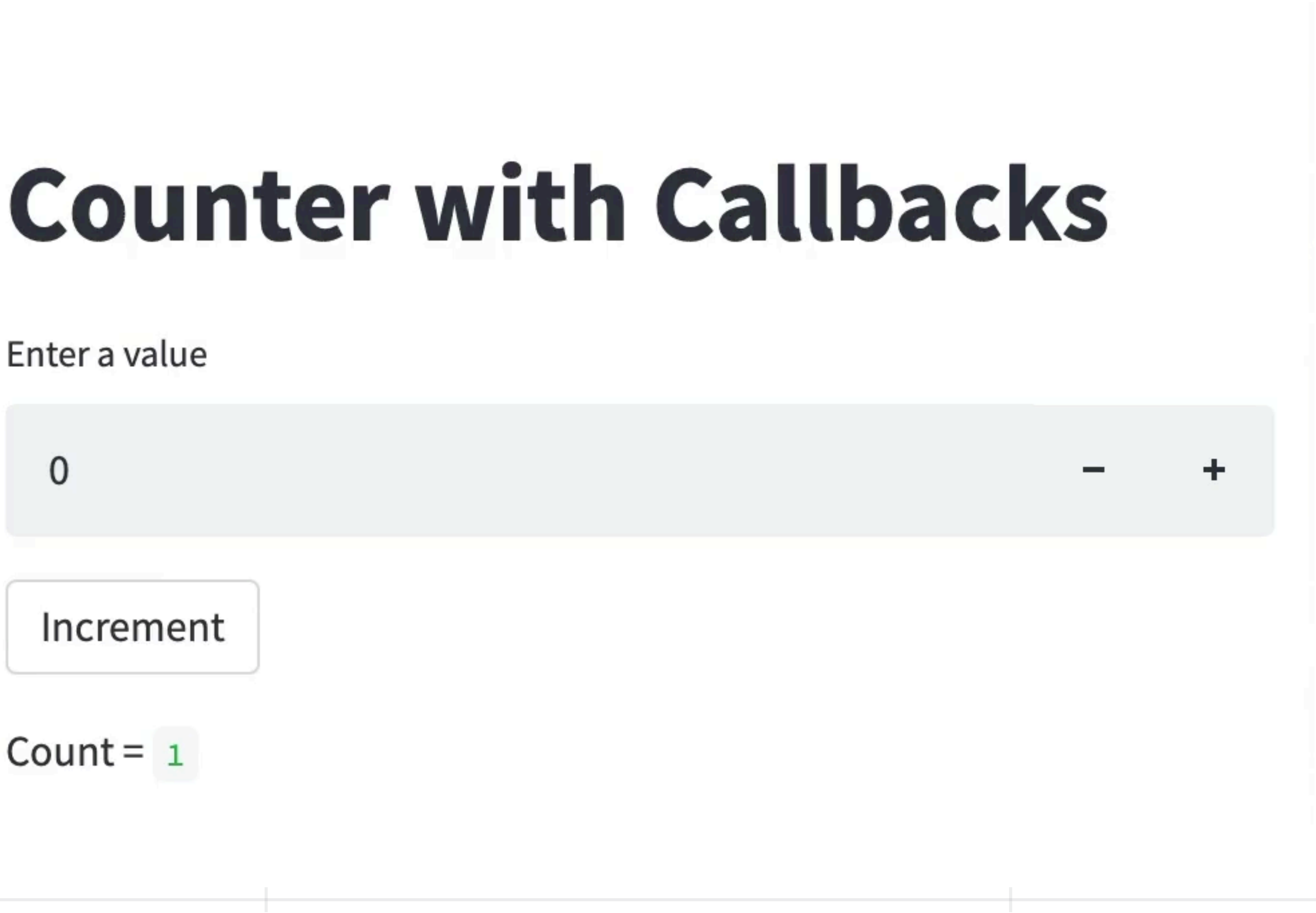
Session state is limited to the current tab in the browser

No permanent storage

Callbacks

```
import streamlit as st

st.title('Counter with Callbacks')
if 'count' not in st.session_state:
    st.session_state.count = 0

increment_value = st.number_input('Enter a value', value=0, step=1)

def increment_counter(increment_value):
    st.session_state.count += increment_value

increment = st.button('Increment', on_click=increment_counter,
    args=(increment_value, ))

st.write('Count = ', st.session_state.count)
```

Callback Movie



Counter with Callbacks

Enter a value

0

Count = 1