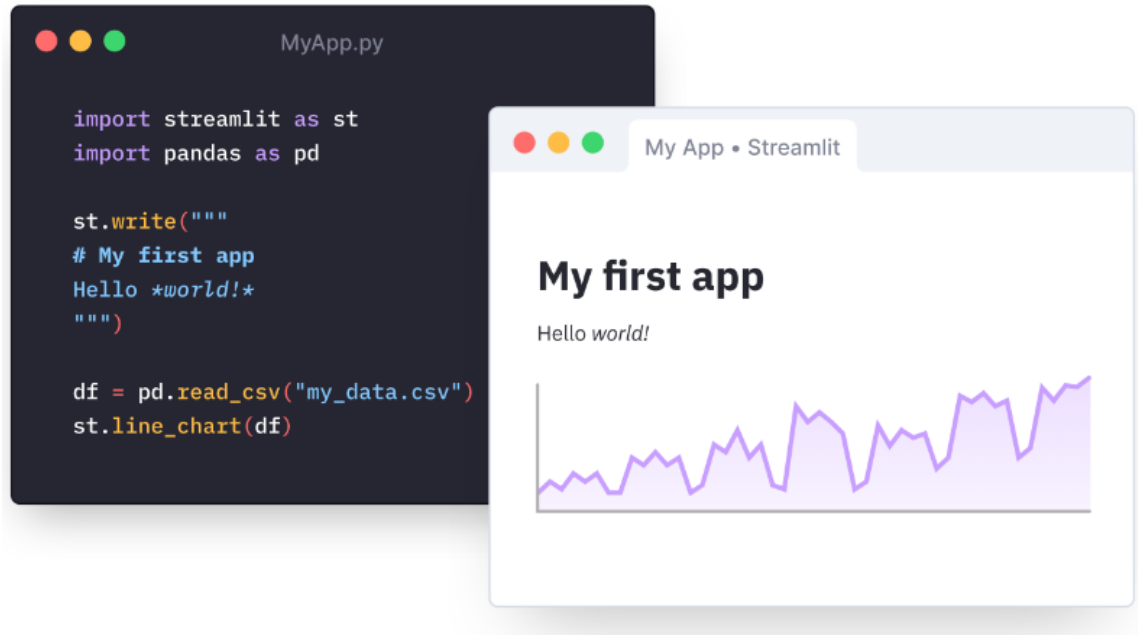
Streamlit을 이용한
Python 시각화 Application 만들기



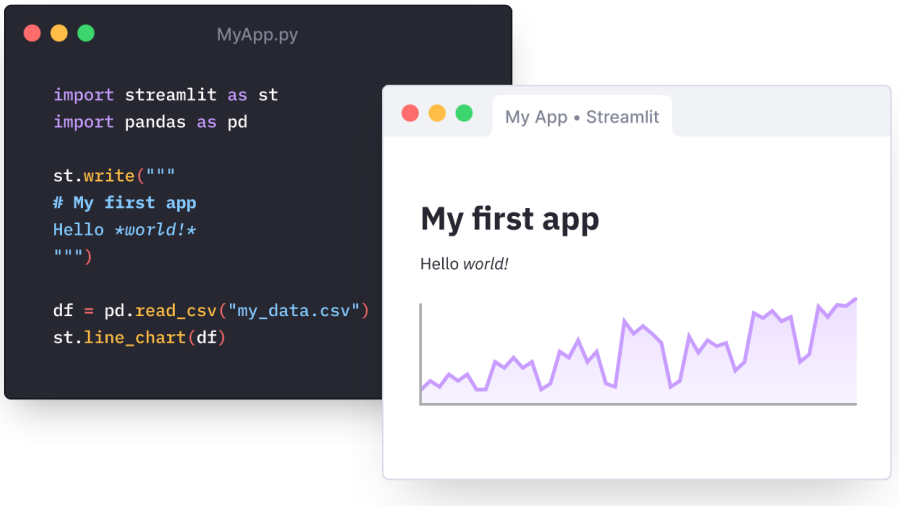
fingerpointlab

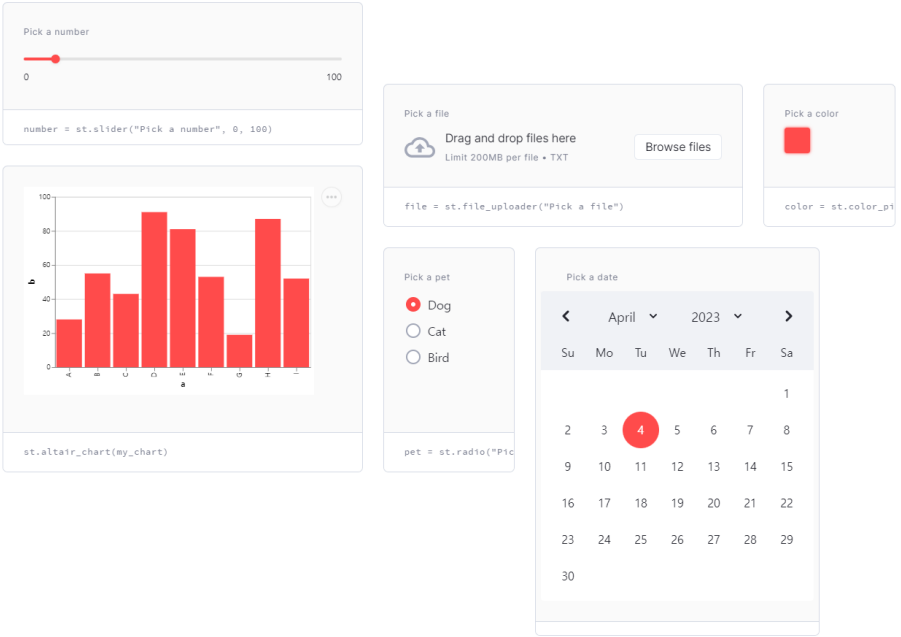
목차

Streamlit 소개

Streamlit 장점

- 간단한 Python code로 웹 기반 시각화 어플리케이션을 만들 수 있음.
- 다양한 위젯을 통한 사용자 입력을 받을 수 있어 Interactive한 데이터 시각화 가능
- 시각화 이외에도 Machine Learning Model Serving 등 Prototype Dashboard 제작에 용이

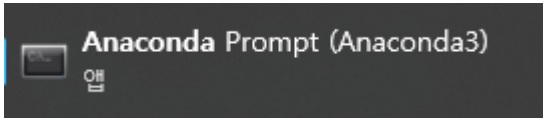




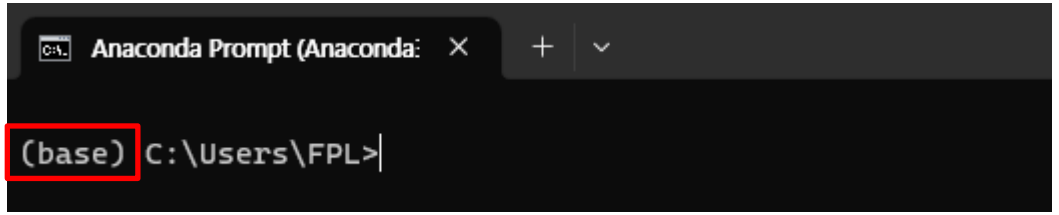
Streamlit 사용 환경 설정하기

1) Anaconda 설치

2) Anaconda prompt 열기



- Anaconda Prompt에서 초기 가상환경이 base 환경인지 확인

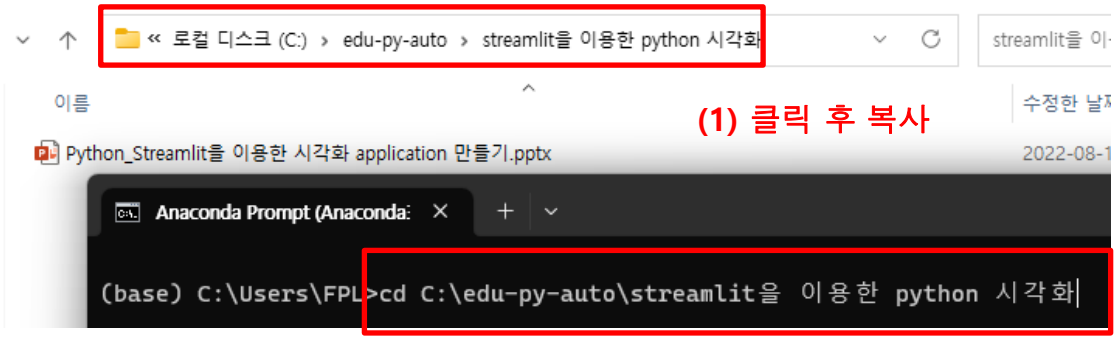


- 콘솔창 내 (base) 가상환경이 활성화 되지 않은 경우

```
conda activate base
```

Streamlit 사용 환경 설정하기(2)

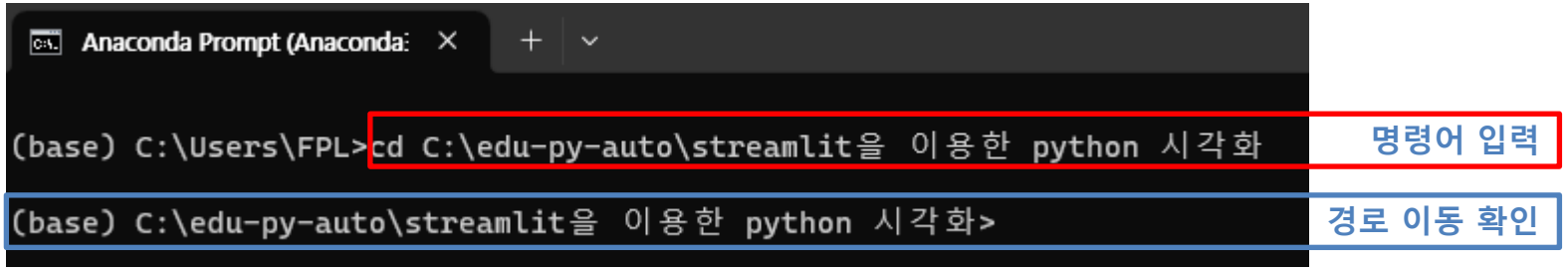
3) 실습 폴더 경로 복사하여 Anaconda Prompt에서 해당 경로로 이동하기



(1) 클릭 후 복사

(2) cd 입력 후 한 칸 띄워서 붙여넣기, Enter

4) Anaconda Prompt 내 경로 이동 확인하기



명령어 입력

경로 이동 확인

Streamlit 사용 환경 설정하기(3)

5) streamlit 패키지 설치하기

```
pip install streamlit
```

```
(base) C:\edu-py-auto\streamlit을 이용한 python 시각화>pip install streamlit
Collecting streamlit
  Downloading streamlit-1.20.0-py2.py3-none-any.whl (9.6 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 9.6/9.6 MB 21.2 MB/s eta 0:00:00
```

- ‘Successfully installed ~’ 라는 문구가 나오면 설치 완료
- Streamlit 구동시 필요한 다른 package(pandas, numpy 등)도 자동으로 설치됨

```
Successfully installed MarkupSafe-2.1.2 altair-4.2.2 attrs-22.2.0 blinker-1.6 cachetools-5.3.0 charset-normalizer-3.1.0
click-8.1.3 gitdb-4.0.10 gitpython-3.1.31 idna-3.4 importlib-metadata-6.1.0 jinja2-3.1.2 jsonschema-4.17.3 markdown-it-p
y-2.2.0 mdurl-0.1.2 numpy-1.24.2 packaging-23.0 pandas-1.5.3 pillow-9.5.0 protobuf-3.20.3 pyarrow-11.0.0 pydeck-0.8.0 py
gments-2.14.0 pympler-1.0.1 pyrsistent-0.19.3 pytz-2023.3 pytz-deprecation-shim-0.1.0.post0 requests-2.28.2 rich-13.3.3
semver-3.0.0 six-1.16.0 smmap-5.0.0 streamlit-1.20.0 toml-0.10.2 toolz-0.12.0 typing-extensions-4.5.0 tzdata-2023.3 tzlo
cal-4.3 urllib3-1.26.15 validators-0.20.0 watchdog-3.0.0 zipp-3.15.0

(base) C:\edu-py-auto\streamlit을 이용한 python 시각화>
```

Streamlit 사용 환경 설정하기(4)

6) plotly 패키지(시각화 라이브러리) 설치하기

```
pip install plotly
```

```
(base) C:\edu-py-auto\streamlit을 이용한 python 시각화>pip install plotly
Collecting plotly
  Using cached plotly-5.14.1-py2.py3-none-any.whl (15.3 MB)
Collecting tenacity>=6.2.0
```

- ‘Successfully installed ~’ 라는 문구가 나오면 설치 완료

Streamlit Application 만들기 및 실행하기

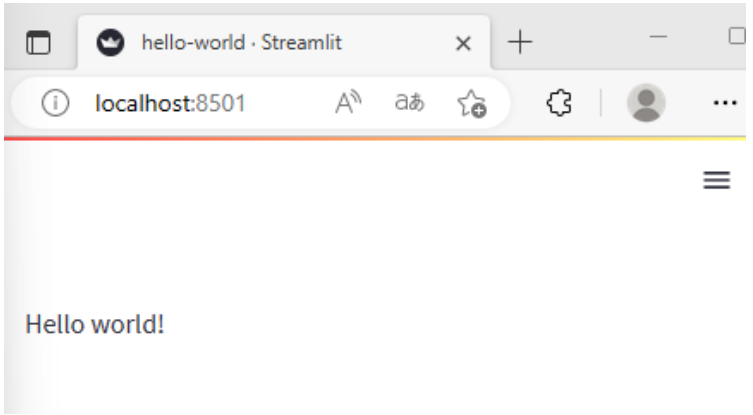
1) hello-world.py 파일 생성 후 다음과 같이 작성하여 저장

```
# hello-world.py 파일

import streamlit as st

st.write('Hello world!')
```

★streamlit 실행 : Terminal(Console창)에서 다음 명령어를 통해 어플리케이션 실행

```
streamlit run hello-world.py
```

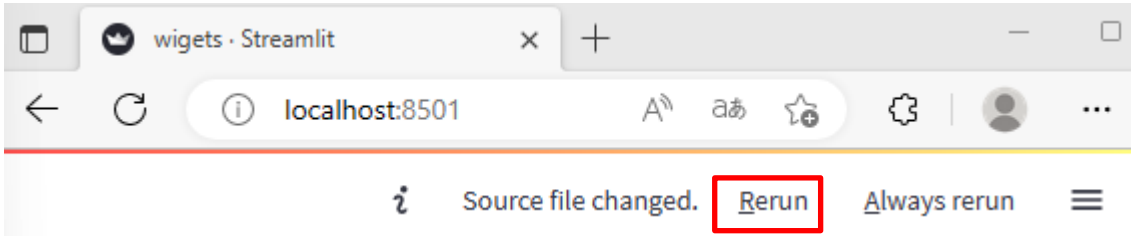
```
PS C:₩edu-py-auto₩streamlit을 이용한 python 시각화> streamlit run hello-world.py

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8501
  Network URL: http://192.168.0.5:8501
```



- **http://localhost:8501이라는 주소로 application이 구동 중. 터미널 창을 닫거나 Ctrl+C를 입력 시 종료되며, 어플리케이션도 종료됨.**

Streamlit Application 만들기 및 실행하기

- **Python code에 변동이 생겼을 경우, streamlit 웹 화면에서 rerun 버튼 클릭하여 변동내용 반영**



- **Anaconda 가상환경에서 설치된 패키지 리스트 확인하기**

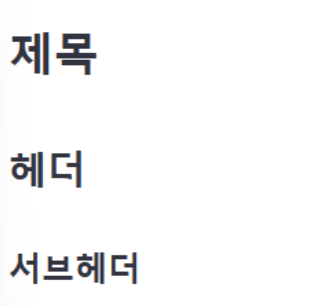
```
conda list
```

```
PS C:₩edu-py-auto₩streamlit을 이용한 python 시각화> conda list
# packages in environment at C:₩Users₩FPL₩Anaconda3:
#
# Name                    Version                   Build  Channel
_ipyw_jlab_nb_ext_conf    0.1.0                     py38_0
alabaster                 0.7.12                 pyhd3eb1b0_0
altair                    4.2.0                     pypi_0    pypi
altgraph                  0.17.2                    pypi_0    pypi
anaconda                  2021.05                   py38_0
anaconda-client           1.7.2                     py38_0
anaconda-navigator        2.0.3                     py38_0
anaconda-project          0.9.1                  pyhd3eb1b0_1
anyio                     3.6.2                     pypi_0    pypi
appdirs                   1.4.4                       py_0
argh                      0.26.2                    py38_0
argon2-cffi               20.1.0                 py38h2bbff1b_1
asn1crypto                1.4.0                       py_0
astroid                   2.5                    py38haa95532_1
```

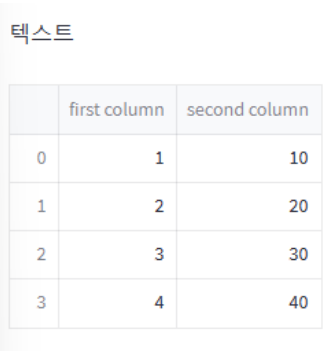
제목과 텍스트 작성하기

제목 및 텍스트, pandas Dataframe 출력

```
# 2-header-and-text.py

# 필요 패키지 불러오기
import streamlit as st
import pandas as pd

# 제목 및 부제목
st.title("제목")
st.header("헤더")
st.subheader("서브헤더")
```




```
# 텍스트
st.write("텍스트")

# 데이터프레임 표현
df = pd.DataFrame({
    'first column': [1, 2, 3, 4],
    'second column': [10, 20, 30, 40]
    })
st.write(df)
```



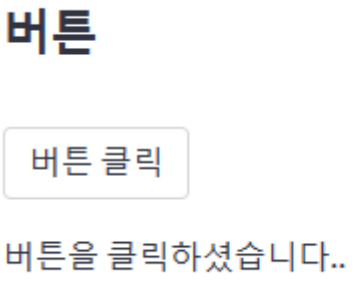
	first column	second column
0	1	10
1	2	20
2	3	30
3	4	40

사용자 입력을 위한 위젯 만들기

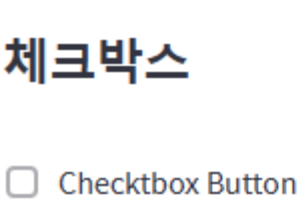
버튼과 체크박스 : st.button(“텍스트”), st.checkbox(“텍스트”)

– 사용자가 클릭한 경우 True를 리턴, if문을 사용하여 클릭 이후의 동작을 정의할 수 있음.

```
# 3-wigets.py

import streamlit as st

st.subheader('버튼')

# 버튼
if st.button("버튼 클릭"):
    st.write("버튼을 클릭하셨습니다..")
```




```
st.subheader('체크박스')

# 체크박스
checkbox_btn = st.checkbox('Checktbox Button')
# 체크박스 선택 시
if checkbox_btn:
    st.write('체크박스를 클릭하셨습니다..')
```

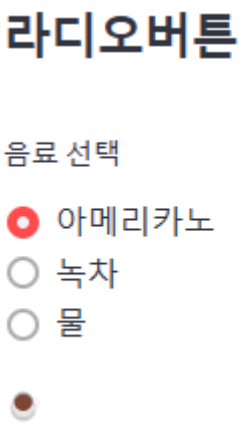


사용자 입력을 위한 위젯 만들기

라디오버튼 : st.radio(“텍스트”, (선택1, 선택2, 선택3, ...))

– 사용자 선택값을 문자열로 return하며, default로 첫번째 항목이 선택됨.

```
# 3-wigets.py

import streamlit as st

st.subheader('라디오버튼')
# 라디오버튼
selected_item = st.radio("음료 선택", ("아메리카노", "녹차", "물"))

if selected_item == "아메리카노":
  st.write("☕")
elif selected_item == "녹차":
  st.write("🍵")
elif selected_item == "물":
  st.write("💧")
```

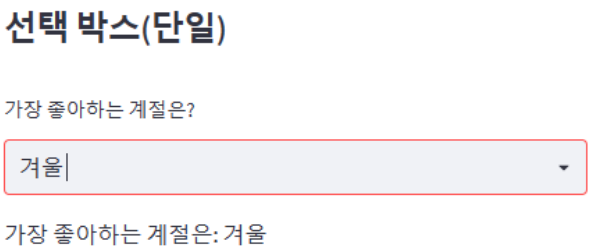


사용자 입력을 위한 위젯 만들기

단일선택 : st.selectbox('텍스트', (선택1, 선택2, 선택3, ...))

- 선택항목을 set으로 정의하며, 사용자 선택값을 string 형식으로 return

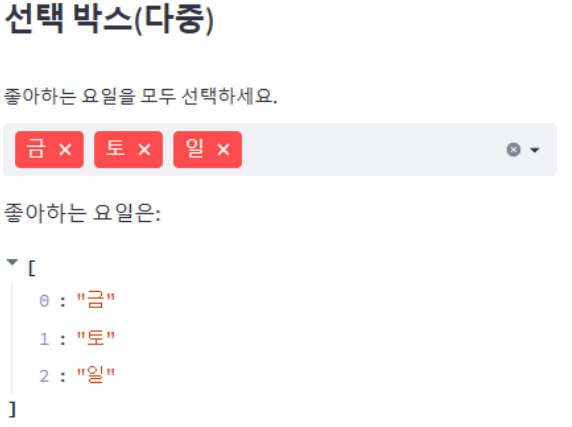
```
# 3-wigets.py

import streamlit as st

# 선택 박스 (단일 선택)
option = st.selectbox('가장 좋아하는 계절은?', ('봄', '여름', '가을', '겨울'))
st.write('가장 좋아하는 계절은:', option)
```



다중선택 : st.multiselect('텍스트', [선택1, 선택2, 선택3, ...])

- 선택항목을 list으로 정의하며, 사용자 선택값을 dictionary 형식으로 return

```
st.subheader('선택 박스(다중)')
# 선택 박스 (다중 선택)
multi_select = st.multiselect('좋아하는 요일을 모두 선택하세요.',
                              ['월', '화', '수', '목', '금', '토', '일'])
st.write('좋아하는 요일은:', multi_select)
```

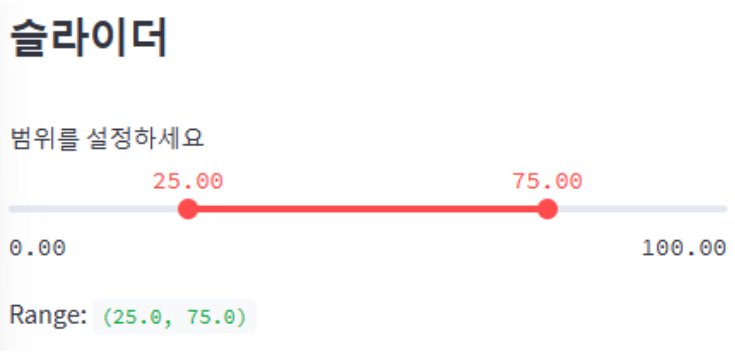


사용자 입력을 위한 위젯 만들기

수치형 범위선택 슬라이더 : st.slider('텍스트', 시작값, 끝값, (default시작값, defautl끝값))

- 사용자 선택 범위값을 set 형식으로 return

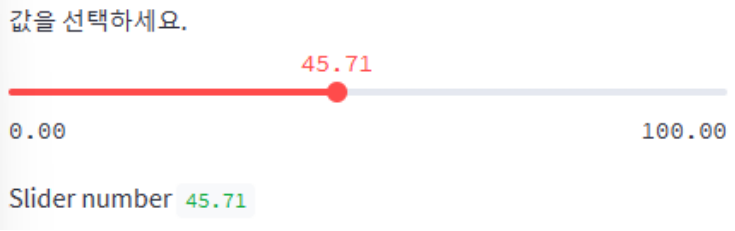
```
# 3-wigets.py

import streamlit as st

st.subheader('슬라이더')
# 수치 데이터 입력을 위한 슬라이더

# 범위 선택 슬라이더
values = st.slider('범위를 설정하세요', 0.0, 100.0, (25.0, 75.0))
st.write('Range:', values)
```



단일 수치 선택 슬라이더 : st.slider('텍스트', 시작값, 끝값)

- 사용자 선택 범위값을 set 형식으로 return

```
# 단일 수치 선택 슬라이더
value = st.slider('값을 선택하세요.', 0.0, 100.0)
st.write('Slider number', value)
```



사용자 입력을 위한 위젯 만들기

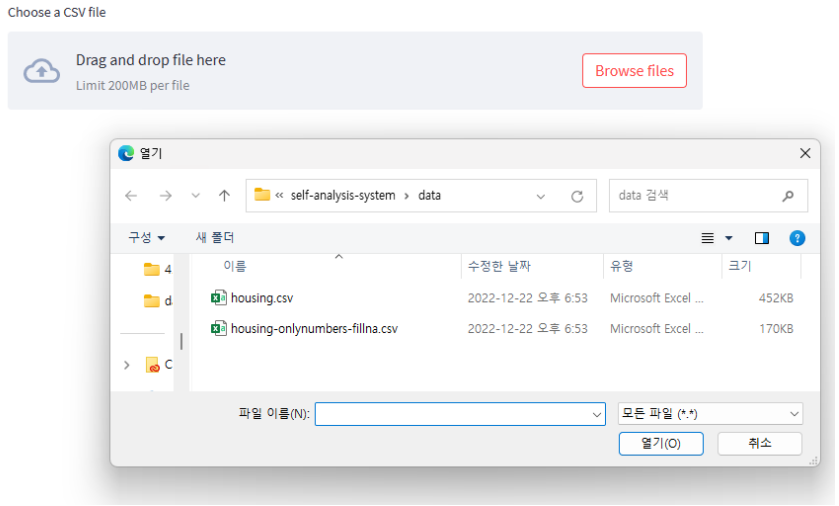
다양한 형식의 데이터 입력

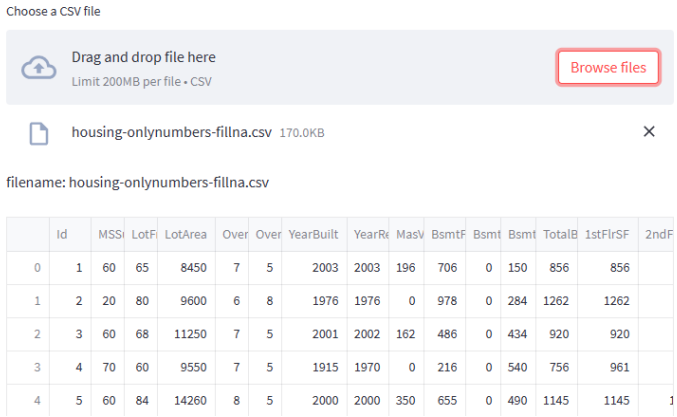
```
# 3-wigets.py

import streamlit as st

st.subheader('다양한 형식의 데이터 입력')
# 단순 텍스트 데이터 입력
txt_input = st.text_input('텍스트를 입력하세요.')

# 여러 줄의 텍스트 입력
txt_input_multi = st.text_area('여러 줄의 텍스트를 입력하세요')

# 수치 데이터 입력
txt_input_number = st.text_input('숫자를 입력하세요')

# 날짜 입력
input_date = st.date_input('날짜를 선택하세요')

# 시간 입력
input_time = st.time_input('시간을 선택하세요')
```

사용자 입력을 위한 위젯 만들기

파일 데이터 입력
– st.file_uploader('텍스트', type=['파일형식1', ..])

```python
# 3_wigets_2.py
import streamlit as st
import pandas as pd

# file uploader - 하나의 파일을 업로드
uploaded_file = st.file_uploader("Choose a CSV file", type=['csv'])

if uploaded_file is not None:
    st.write("filename:", uploaded_file.name)
    df = pd.read_csv(uploaded_file)
    st.write(df)

```





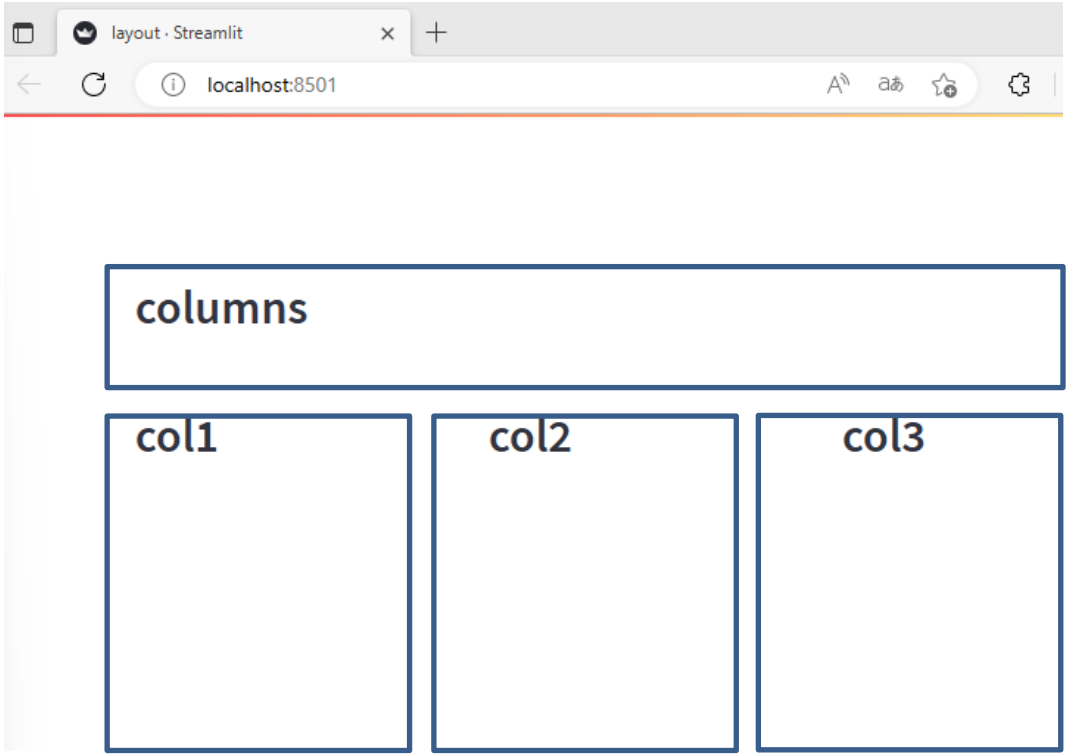
	Id	MSS	LotF	LotArea	Over	Over	YearBuilt	YearR	MasV	BsmtF	Bsmt	Bsmt	TotalB	1stFlrSF	2ndF
0	1	60	65	8450	7	5	2003	2003	196	706	0	150	856	856	
1	2	20	80	9600	6	8	1976	1976	0	978	0	284	1262	1262	
2	3	60	68	11250	7	5	2001	2002	162	486	0	434	920	920	
3	4	70	60	9550	7	5	1915	1970	0	216	0	540	756	961	
4	5	60	84	14260	8	5	2000	2000	350	655	0	490	1145	1145	

Layout 구성하기

Column layout 구성하기 : st.columns(컬럼수)

- 컬럼 수와 동일한 개수의 변수 선언 필요
- with문을 이용하여 각 컬럼 레이아웃의 내용 작성
- code 상에서 위에서 아래로 구문이 순차적으로 실행되므로, 아래 예시에서 layout 구성은 4개의 영역으로 나뉘어짐.

```python
# 4_layout_1.py
import streamlit as st

st.header('columns')

# column layout
col1, col2, col3 = st.columns(3)

with col1:
    st.header('col1')

with col2:
    st.header('col2')

with col3:
    st.header('col3')


```



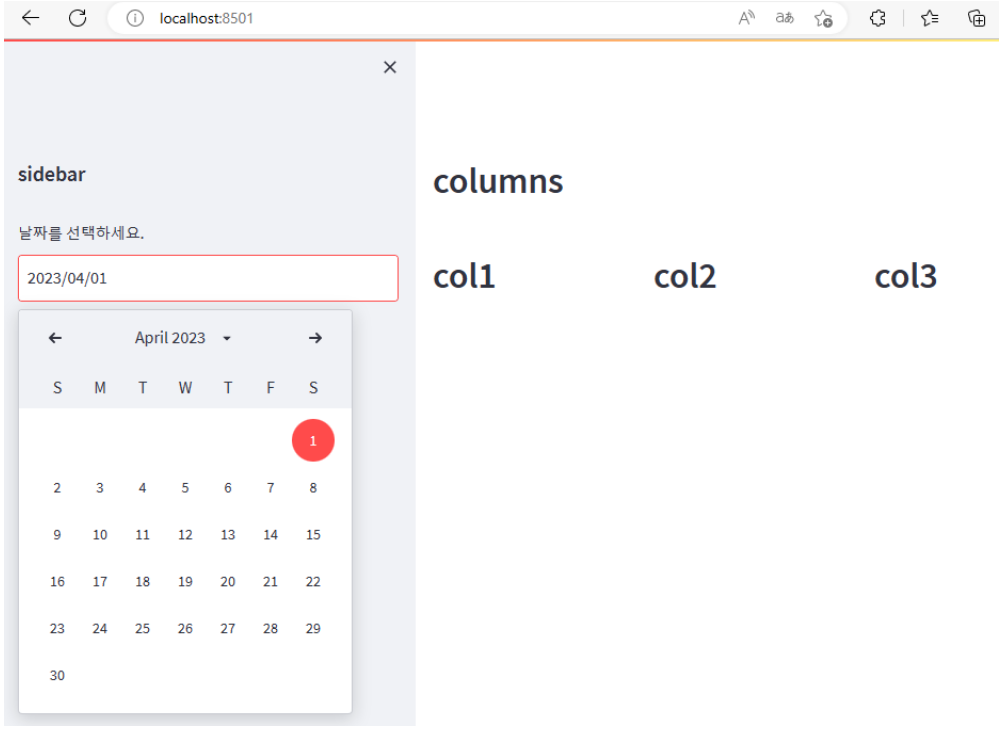
Layout 구성하기

Sidebar layout 구성하기 : st.sidebar.메소드명

- 사이드바는 화면상 좌측에 위치하며, 화면상에서 숨기기 가능
- streamlit에선 주로 사용자 입력을 사이드바에 위치시킴
- 메인 화면에선 st.메소드명을 기입하여 메인화면에 출력결과나 위젯 등을 구현했다면, 사이드바는 st.sidebar.메소드명을 통해 사이드바에도 동일하게 구현 가능

4_layout_1.py

```
19 # sidebar
20 st.sidebar.header('sidebar')
21
22 st.sidebar.date_input('날짜를 선택하세요.')
```

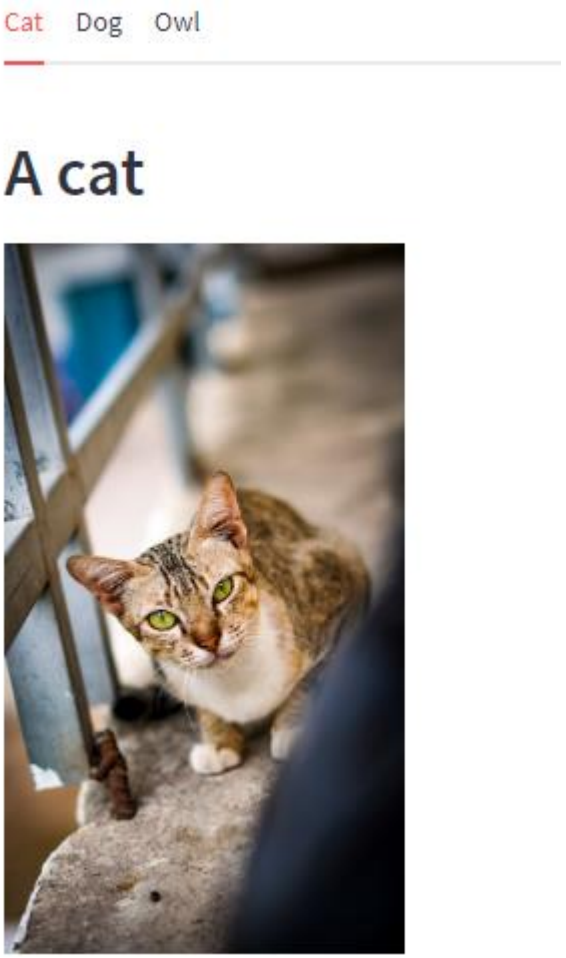


Layout 구성하기

tab layout 구성하기 : st.tabs(['탭1', '탭2', '탭3'])

- (추가) 이미지 출력하기 : st.image("이미지주소")

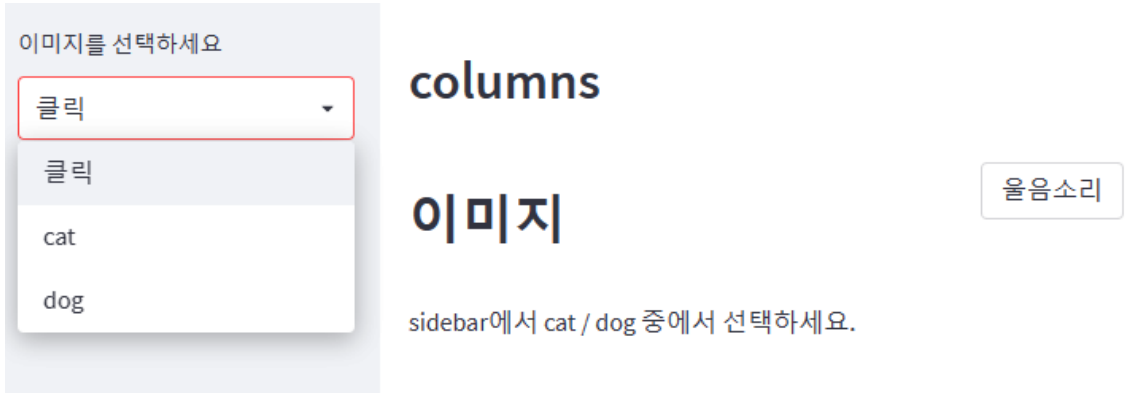
```python
# 4_layout_2.py

import streamlit as st

tab1, tab2, tab3 = st.tabs(["Cat", "Dog", "Owl"])

with tab1:
    st.header("A cat")
    st.image("https://static.streamlit.io/examples/cat.jpg", width=200)

with tab2:
    st.header("A dog")
    st.image("https://static.streamlit.io/examples/dog.jpg", width=200)

with tab3:
    st.header("An owl")
    st.image("https://static.streamlit.io/examples/owl.jpg", width=200)
```



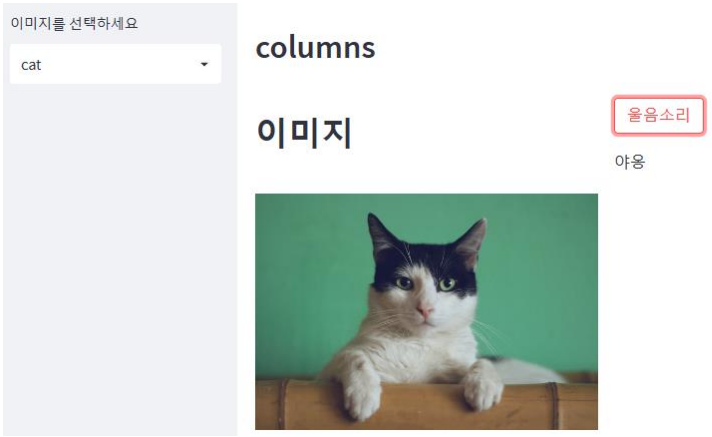
Layout 구성하기 - 실습

Sidebar layout 구성 실습

- 초기화면



- Sidebar에서 'cat' 선택 후 '울음소리' 버튼 클릭



- Sidebar에서 'dog' 선택 후 '울음소리' 버튼 클릭

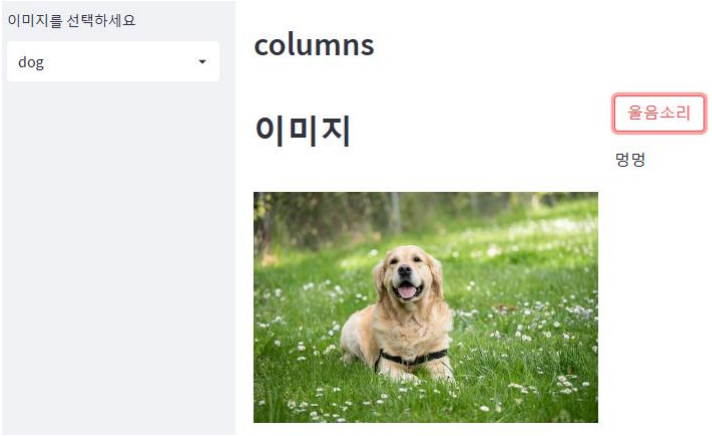
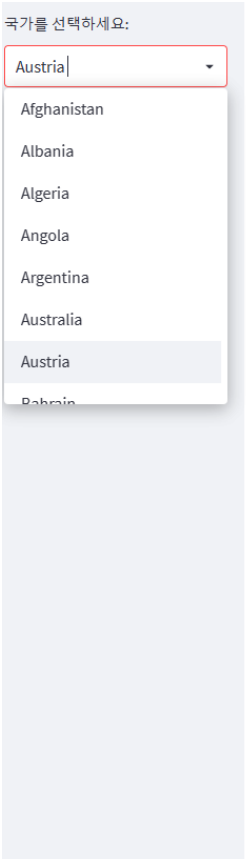


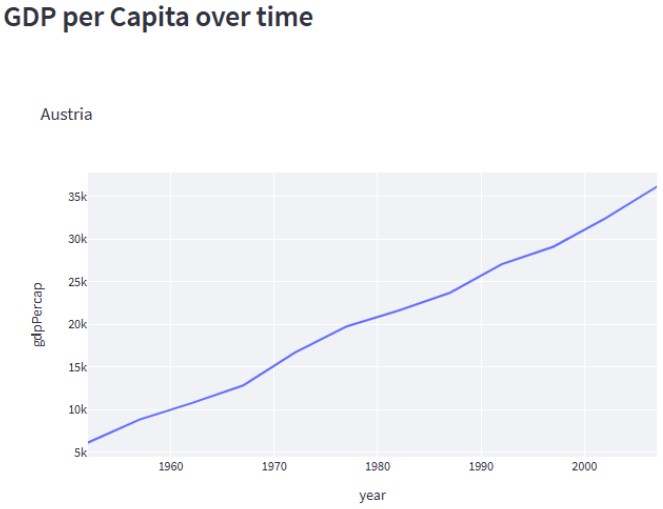
차트 표현하기

Plotly를 이용한 라인차트 구현 : px.line(dataset, x='컬럼명', y='컬럼명', title='타이틀')

- 예시데이터: gdp_by_country.csv (국가-연도별 GDP, 인구수 데이터)

```python
# 5_chart_1.py
import streamlit as st
import pandas as pd
import plotly.express as px

# 데이터 불러오기
data = 'data/gdp_by_country.csv'
df = pd.read_csv(data)

# 원본 데이터에서 국가명 리스트 가져오기
clist = df['country'].unique()

# 사이드바에서 국가명 선택
country = st.sidebar.selectbox("국가를 선택하세요:",clist)

# 메인화면
st.subheader('GDP per Capita over time')

# 선택한 국가명에 따라 연도별 gdp 추이를 라인 그래프 설정하기
fig = px.line(df[df['country'] == country],
    x = "year", y = "gdpPercap", title = country)

# plotly 그래프 출력하기
st.plotly_chart(fig)


# 원본 데이터에서 선택한 국가의 rawdata 확인하기
st.subheader('data')
st.write(df[df['country'] == country])
```





data

	country	continent	year	lifeExp	pop	gdpPercap	iso_alpha	iso_
72	Austria	Europe	1952	66.8000	6927772	6,137.0765	AUT	40
73	Austria	Europe	1957	67.4800	6965860	8,842.5980	AUT	40
74	Austria	Europe	1962	69.5400	7129864	10,750.7211	AUT	40
75	Austria	Europe	1967	70.1400	7376998	12,834.6024	AUT	40

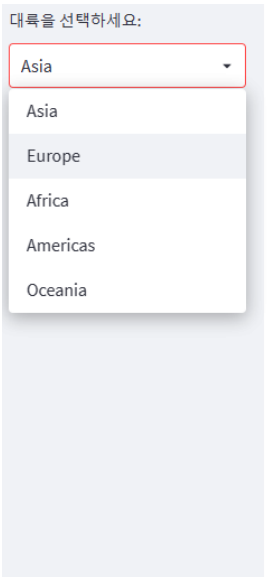
차트 표현하기

Plotly를 이용한 막대차트 구현 : px.bar(dataset, x='컬럼명', y='컬럼명', color='컬럼명')

- **예시데이터: gdp_by_country.csv (국가-연도별 GDP, 인구수 데이터)**

- **plotly express method의 필수옵션 : dataset, x축 컬럼명, y축 컬럼명**
  : 이외 title, color 등은 생략 가능하나 표현하고 싶은 정보는 옵션명을 확인하여 표현

```python
# 5_chart_2.py
import streamlit as st
import pandas as pd
import plotly.express as px

# 데이터 불러오기
data = 'data/gdp_by_country.csv'
df = pd.read_csv(data)

# 원본 데이터에서 대륙명 리스트 가져오기
ctnlist = df['continent'].unique()

# 사이드바에서 대륙 선택
ctn_selected = st.sidebar.selectbox("대륙을 선택하세요:",ctnlist)

# 원본 데이터셋에서 필터링하기
df_filtered = df[df['continent'] == ctn_selected]

# 막대그래프 그리기
st.header('대륙별 인구변화 추이')
fig = px.bar(df_filtered, x='year', y='pop', color='country')
st.plotly_chart(fig)
```



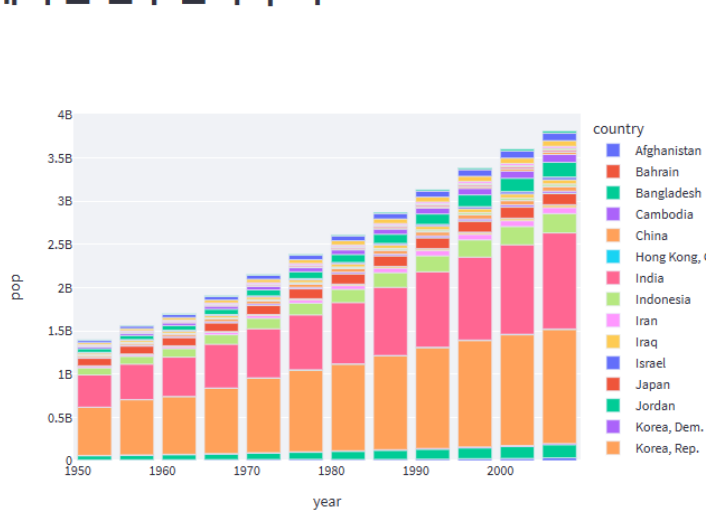


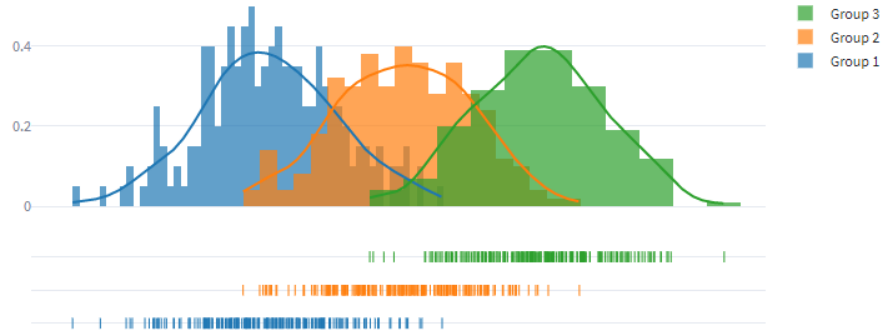
차트 표현하기

Plotly를 이용한 히스토그램 구현 : px.bar(dataset, x='컬럼명', y='컬럼명', color='컬럼명')

- 예시데이터: gdp_by_country.csv (국가-연도별 GDP, 인구수 데이터)

- plotly express method의 필수옵션 : dataset, x축 컬럼명, y축 컬럼명
: 이외 title, color 등은 생략 가능하나 표현하고 싶은 정보는 옵션명을 확인하여 표현

```python
# 5_chart_3.py
import streamlit as st
import numpy as np
import plotly.figure_factory as ff

# 임의의 histogram data 생성
x1 = np.random.randn(200) - 2
x2 = np.random.randn(200)
x3 = np.random.randn(200) + 2

# 데이터를 하나의 리스트로 묶기
hist_data = [x1, x2, x3]
group_labels = ['Group 1', 'Group 2', 'Group 3']

# 데이터의 빈도를 나타내는 히스토그램 차트 생성하기
fig = ff.create_distplot(
        hist_data, group_labels, bin_size=[.1, .25, .5])

# 차트 출력
st.plotly_chart(fig, use_container_width=True)
```

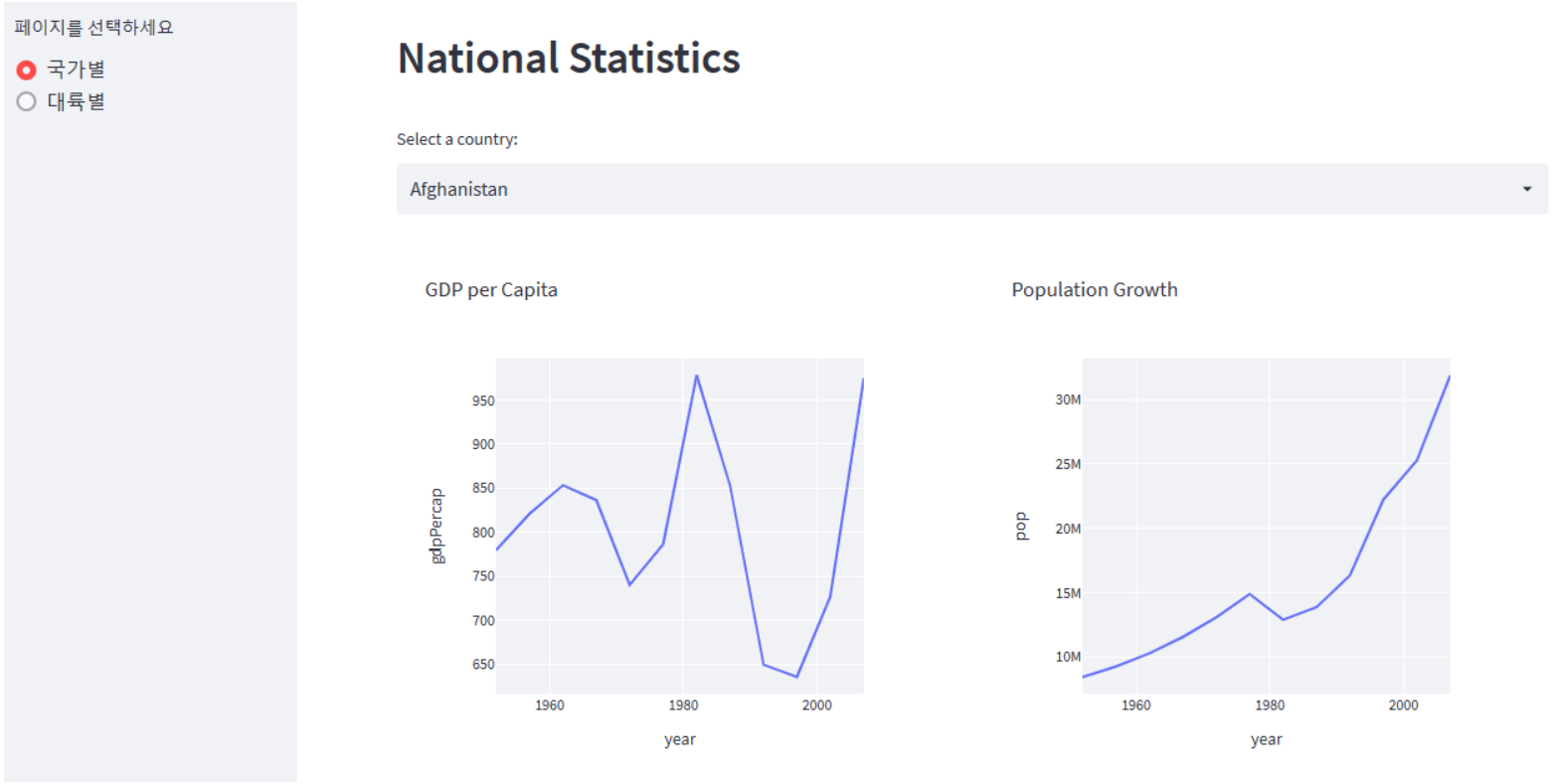


실습과제1 – 시각화 대시보드

Streamlit과 plotly express를 이용하여 국가별/대륙별 GDP, 인구수 추이 시각화 대시보드 만들기

1) 사이드바에서 '국가별' 라디오버튼을 선택한 경우

- 메인화면에서 국가명을 선택할 수 있는 select box 구현
- select box에서 선택한 국가에 따라 GDP 추이와 인구수 추이를 column layout을 이용하여 라인 그래프로 나타내기

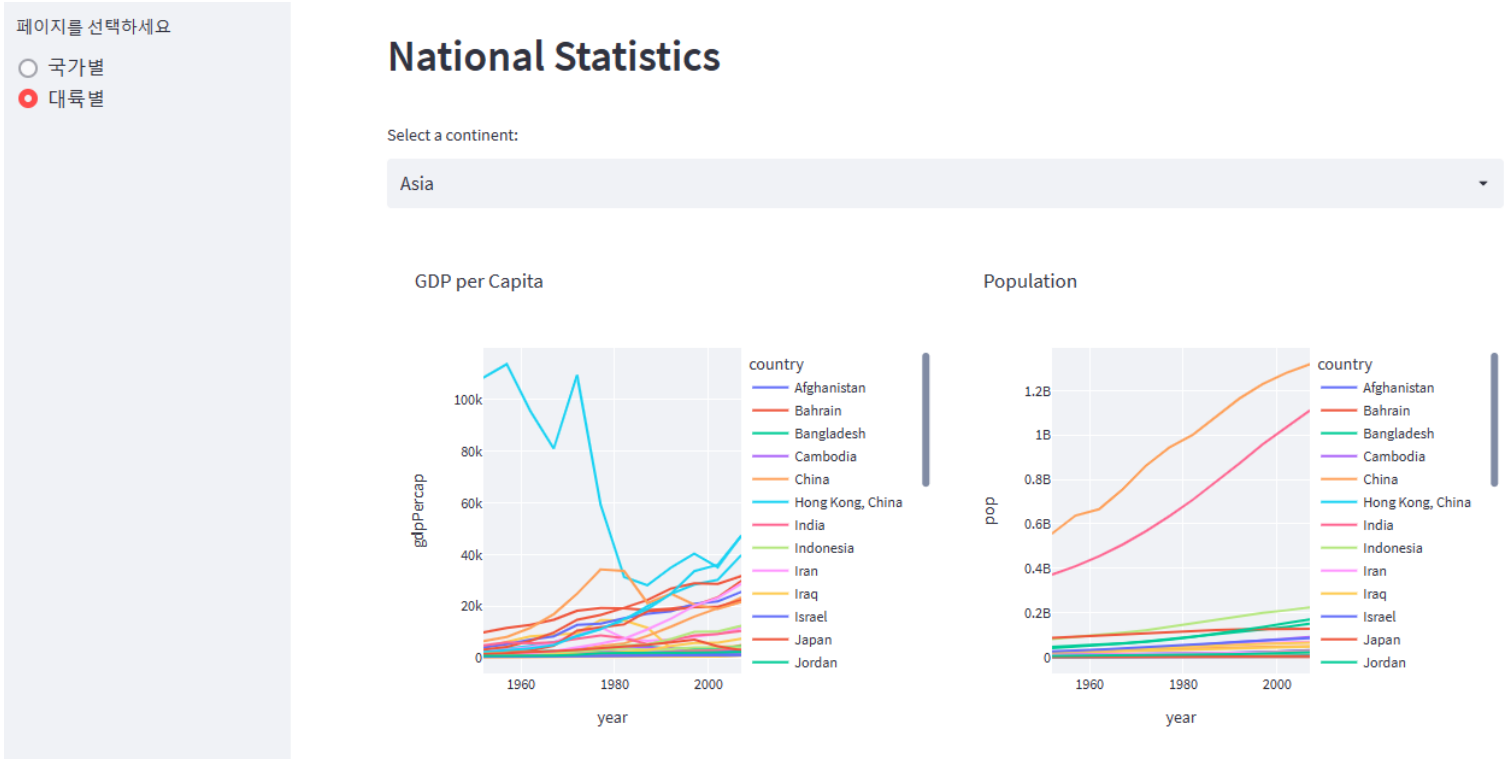


실습과제1 – 시각화 대시보드

Streamlit과 plotly express를 이용하여 국가별/대륙별 GDP, 인구수 추이 시각화 대시보드 만들기

2) 사이드바에서 '대륙별' 라디오버튼을 선택한 경우

- 메인화면에서 대륙명을 선택할 수 있는 select box 구현
- select box에서 선택한 대륙에 따라 국가별 GDP 추이와 인구수 추이를 column layout을 이용하여 라인 그래프로 나타내기



실습과제1 – 시각화 대시보드

```python
# 5_chart_실습과제.py
import streamlit as st
import pandas as pd
import plotly.express as px
st.set_page_config(layout = "wide")

# 데이터 불러오기
data = 'data/gdp_by_country.csv'
df = pd.read_csv(data)

# 사이드바
page = st.sidebar.radio("페이지를 선택하세요", ('국가별','대륙별'))

# 메인페이지
st.header("National Statistics")
if page == '국가별':
    ## Countries
    clist = df['country'].unique()
    country = st.selectbox("Select a country:",clist)
    col1, col2 = st.columns(2)
    fig = px.line(df[df['country'] == country],
                  x = "year", y = "gdpPercap",title = "GDP per Capita")

    col1.plotly_chart(fig,use_container_width = True)
    fig = px.line(df[df['country'] == country],
                  x = "year", y = "pop",title = "Population Growth")

    col2.plotly_chart(fig,use_container_width = True)
else:
    ## Continents
    contlist = df['continent'].unique()

    continent = st.selectbox("Select a continent:",contlist)
    col1,col2 = st.columns(2)
    fig = px.line(df[df['continent'] == continent],
                  x = "year", y = "gdpPercap",
    title = "GDP per Capita",color = 'country')

    col1.plotly_chart(fig)
    fig = px.line(df[df['continent'] == continent],
                  x = "year", y = "pop",
    title = "Population",color = 'country')

    col2.plotly_chart(fig, use_container_width = True)
```

(실습과제 code) Streamlit과 plotly expres를 이용하여 국가별/대륙별 GDP, 인구수 추이 시각화 대시보드 만들기

부가설명

5: st.set_page_config(layout = 'wide')
- 메인화면을 넓게 볼 수 있는 wide 옵션 세팅

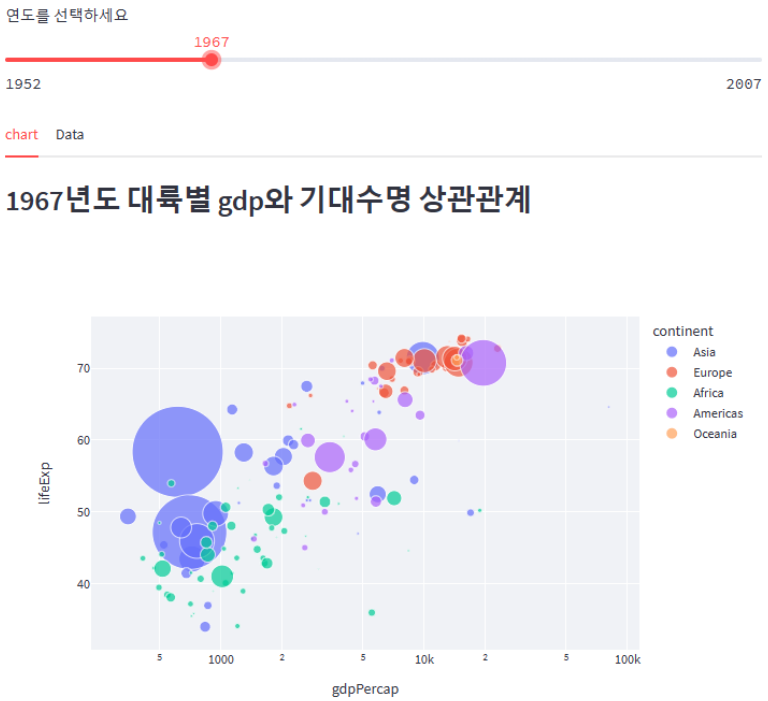
24, 28, 44: use_container_width = True
- 컬럼 넓이에 맞추어 차트 가로 길이를 조절하는 옵션

실습과제2 – 시각화 대시보드

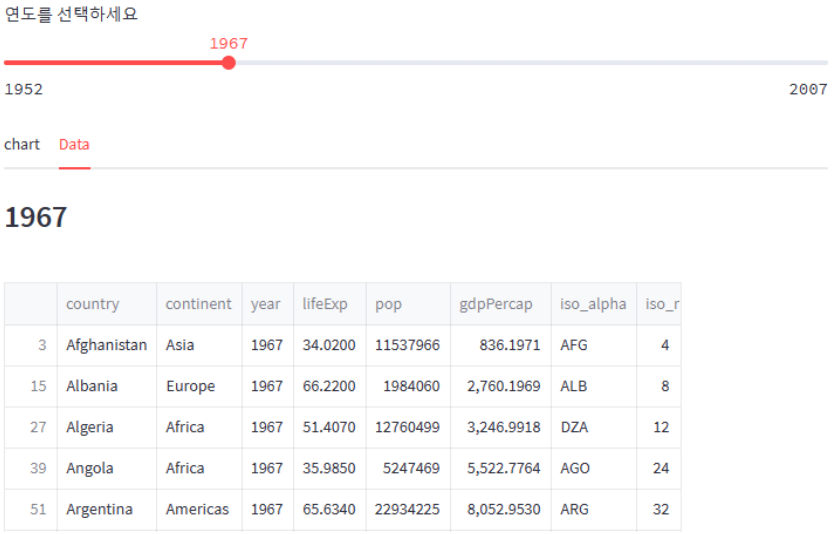
Streamlit과 plotly express를 이용하여
연도-대륙별 GDP와 인구수 상관관계를 interactive하게 나타내는 시각화 대시보드 만들기

1) Slider를 이용하여 사용자가 연도 데이터를 선택
2) 1)에서 선택된 연도 데이터를 기준으로 데이터셋을 필터링하여, scatter chart와 dataframe으로 출력
3) 차트와 데이터는 tab으로 구분되도록 레이아웃을 구성함

- 'chart' 탭을 클릭한 경우



- 'data' 탭을 클릭한 경우



	country	continent	year	lifeExp	pop	gdpPercap	iso_alpha	iso_r
3	Afghanistan	Asia	1967	34.0200	11537966	836.1971	AFG	4
15	Albania	Europe	1967	66.2200	1984060	2,760.1969	ALB	8
27	Algeria	Africa	1967	51.4070	12760499	3,246.9918	DZA	12
39	Angola	Africa	1967	35.9850	5247469	5,522.7764	AGO	24
51	Argentina	Americas	1967	65.6340	22934225	8,052.9530	ARG	32

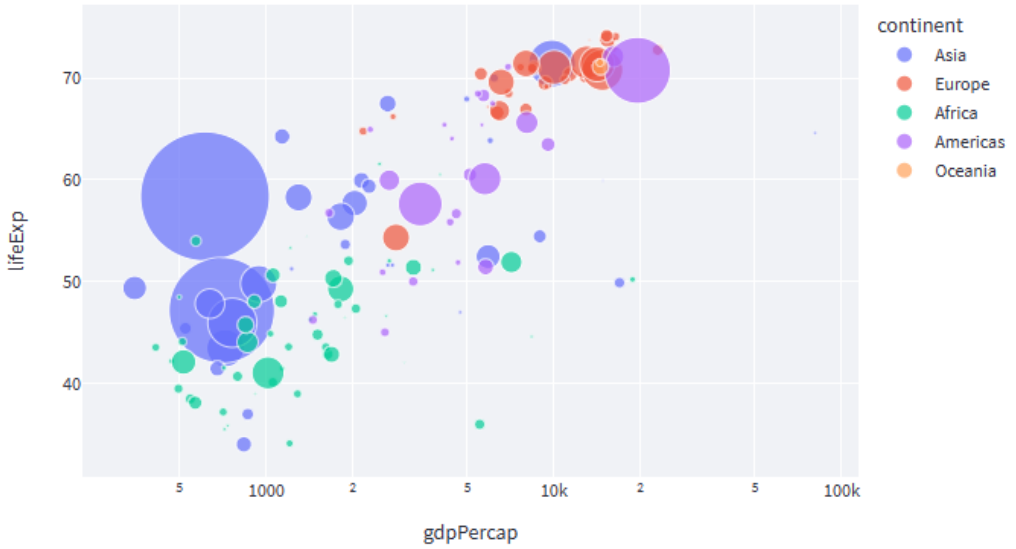
실습과제2 – 시각화 대시보드

Streamlit과 plotly express를 이용하여
연도-대륙별 GDP와 인구수 상관관계를 interactive하게 나타내는 시각화 대시보드 만들기

4) Scatter chart : px.scatter(dataset, x=, y=)

필수 옵션값
- **Dataset : 원본데이터에서 연도별로 필터링된 데이터셋**
- **x : GDP**(gdpPercap)
- **y : 기대수명**(lifeExp)

선택 옵션값
- **size : 인구수**(pop)
- **color : 대륙**(continent)
- log_x : x축 데이터에 로그값을 취함(True)
- size_max : 점(scatter)의 크기 제한



실습과제2 – 시각화 대시보드

```python
# 5_chart_실습과제2.py
import streamlit as st
import pandas as pd
import plotly.express as px

# 데이터 불러오기
data = 'data/gdp_by_country.csv'
df = pd.read_csv(data)

# 연도 리스트
yearlist = df['year'].unique()

# 연도 선택
input_year = st.select_slider('연도를 선택하세요', options=yearlist)

# dataset 필터링
dataset = df.query(f"year=={input_year}")

# chart 설정
fig = px.scatter(
    dataset,
    x="gdpPercap",
    y="lifeExp",
    size="pop",
    color="continent",
    hover_name="country",
    log_x=True,
    size_max=60,
)

tab1, tab2 = st.tabs(["chart", "Data"])
with tab1:
    st.subheader(f"{input_year}년도 대륙별 gdp와 기대수명 상관관계")
    # chart 출력
    st.plotly_chart(fig, use_container_width=True)
with tab2:
    st.subheader(f"{input_year}")
    # dataset 출력
    st.write(dataset)
```

(실습과제 code)
Streamlit과 plotly express를 이용하여 연도-대륙별 GDP와 인구수 상관관계를 interactive하게 나타내는 시각화 대시보드 만들기

과제 – 국가별 CO2 배출량 연도별 추이 Dashboard

Streamlit과 plotly express를 이용하여
연도-국가별 CO2 배출량을 interactive하게 나타내는 시각화 대시보드 만들기

- 데이터셋 (co2.csv)
  - Entity : 국가명(영문)
  - Code : 국가코드
  - Year : 연도
  - Annual CO2 Emissions : 연도별 이산화탄소 배출량

	Entity	Code	Year	Annual CO_2 emissions
30926	South Korea	KOR	1914	25648.0
30927	South Korea	KOR	1915	32012.0
30928	South Korea	KOR	1916	26612.0
30929	South Korea	KOR	1917	
30930	South Korea	KOR	1918	26227.0
30931	South Korea	KOR	1919	31048.0
30932	South Korea	KOR	1920	40304.0
30933	South Korea	KOR	1921	43389.0
30934	South Korea	KOR	1922	44354.0

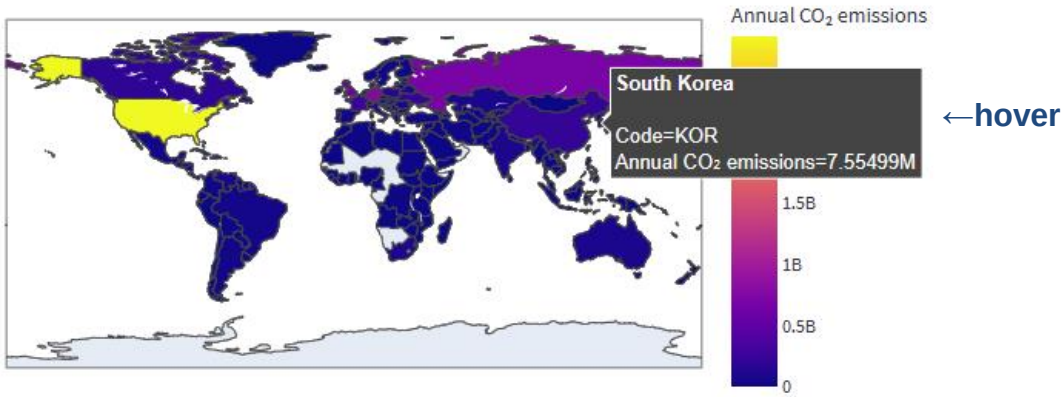
과제 – 국가별 CO2 배출량 연도별 추이 Dashboard

Streamlit과 plotly express를 이용하여
연도-국가별 CO2 배출량을 interactive하게 나타내는 시각화 대시보드 만들기

[별첨] 국가코드 데이터를 이용하여 세계지도에 나타내기
- **px.choropleth(dataset, locations='Code', color)**
- **주의: dataset에 국가코드 데이터가 있어야만 사용가능**

```
# Create a choropleth map using Plotly Express
fig = px.choropleth(data_frame=data,
                    locations='Code',
                    color='Annual CO₂ emissions',
                    hover_name='Entity',
                    title=f'CO₂ Emissions Map ({selected_year})')
st.plotly_chart(fig)
```

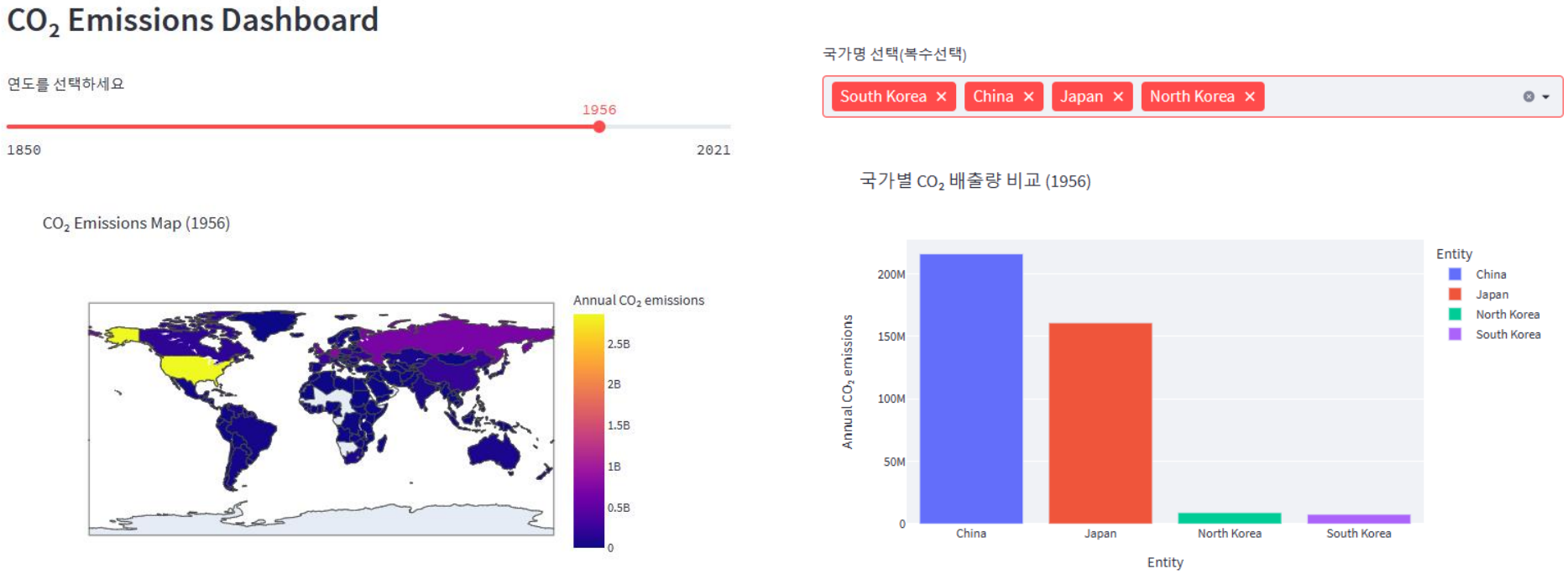
CO_2 Emissions Map (1956) ←title



←hover

과제 – 국가별 CO2 배출량 연도별 추이 Dashboard

Streamlit과 plotly express를 이용하여
연도-국가별 CO2 배출량을 interactive하게 나타내는 시각화 대시보드 구현 예시



자주 나타나는 에러 유형 해결방법

구분	에러 유형	해결방법
Excel 파일을 불러올때 pd.read_excel() 에러	**Openpyxl 모듈이 존재하지 않음 (Module not found error)**	terminal에서 openpyxl 설치 **$ pip install openpyxl**
	'utf-8' 에러	pd.read_excel('file.xlsx', **encoding='cp949'**)
시각화 관련 라이브러리 (matplotlib, seaborn, plotly 등) 자주 발생하는 에러	**Dataframe에 결측치 존재**	결측치가 존재하는 row/column 제외 or 결측치 보간
	수치형 데이터 중 결측 데이터가 문자열('-')로 표현되어 있음	수치형으로 보간

End of Document