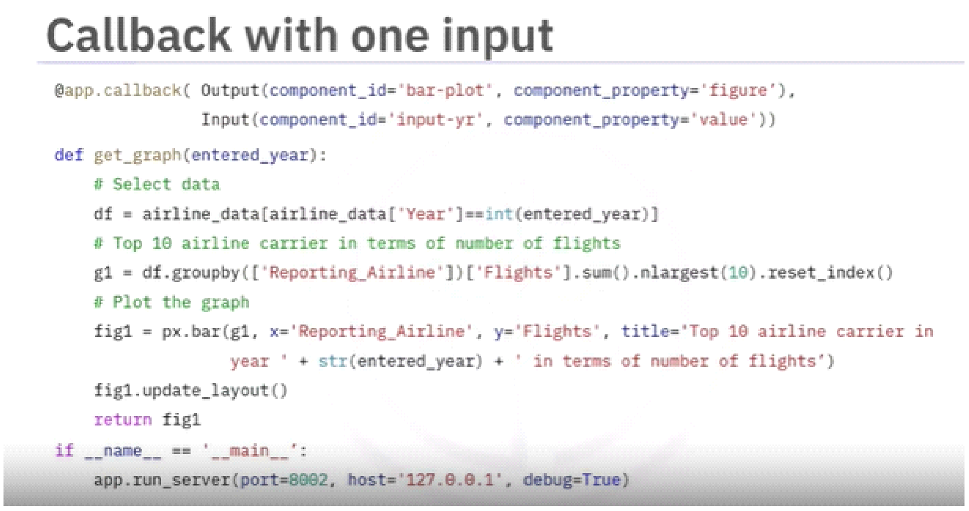
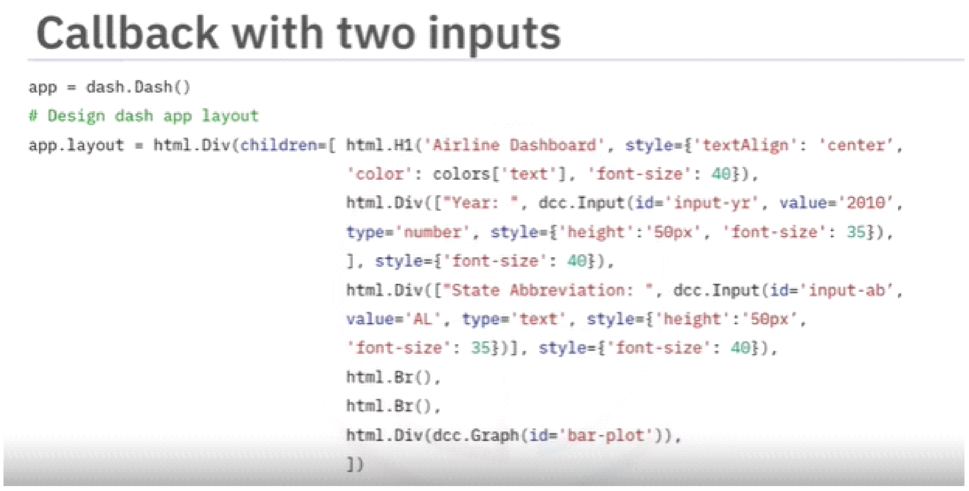
Visualization with Python

2024年2月2日　　22:47



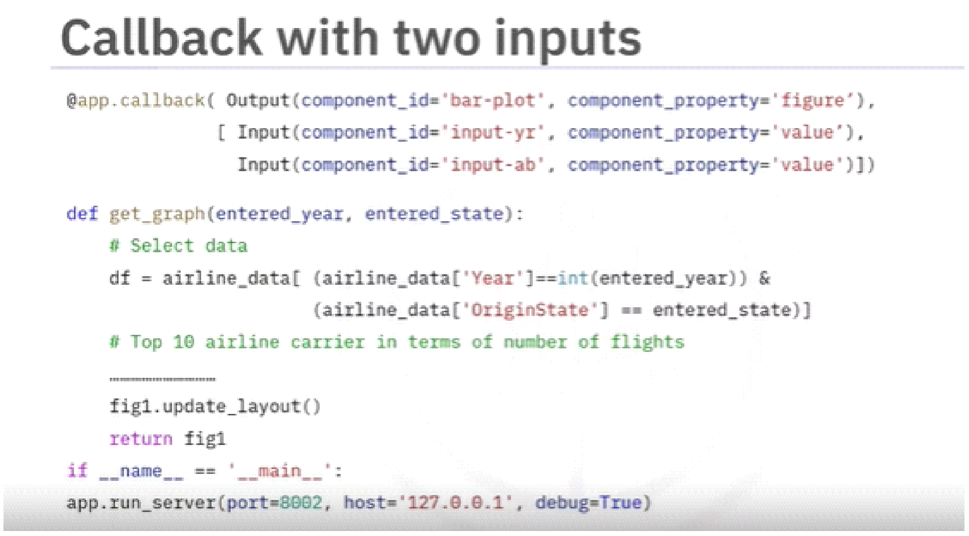
Callback with one input

```
@app.callback( Output(component_id='bar-plot', component_property='figure'),
               Input(component_id='input-yr', component_property='value'))
def get_graph(entered_year):
    # Select data
    df = airline_data[airline_data['Year']==int(entered_year)]
    # Top 10 airline carrier in terms of number of flights
    g1 = df.groupby(['Reporting_Airline'])['Flights'].sum().nlargest(10).reset_index()
    # Plot the graph
    fig1 = px.bar(g1, x='Reporting_Airline', y='Flights', title='Top 10 airline carrier in
                 year ' + str(entered_year) + ' in terms of number of flights')
    fig1.update_layout()
    return fig1
if __name__ == '__main__':
    app.run_server(port=8002, host='127.0.0.1', debug=True)
```



Callback with two inputs

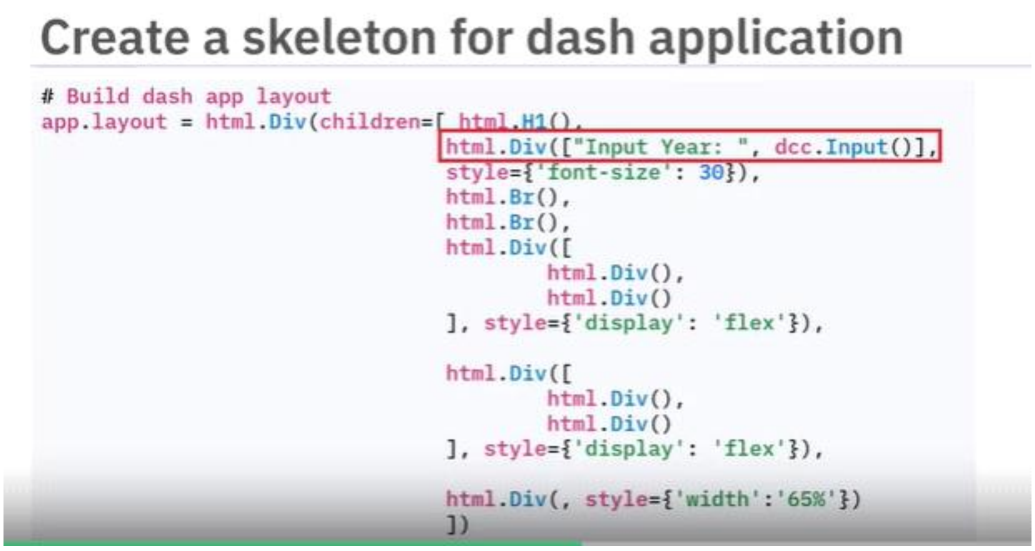
```
app = dash.Dash()
# Design dash app layout
app.layout = html.Div(children=[ html.H1('Airline Dashboard', style={'textAlign': 'center',
                                'color': colors['text'], 'font-size': 40}),
                                html.Div(["Year: ", dcc.Input(id='input-yr', value='2010',
                                type='number', style={'height':'50px', 'font-size': 35}),
                                ], style={'font-size': 40}),
                                html.Div(["State Abbreviation: ", dcc.Input(id='input-ab',
                                value='AL', type='text', style={'height':'50px',
                                'font-size': 35})], style={'font-size': 40}),
                                html.Br(),
                                html.Br(),
                                html.Div(dcc.Graph(id='bar-plot')),
                                ])
```



Callback with two inputs

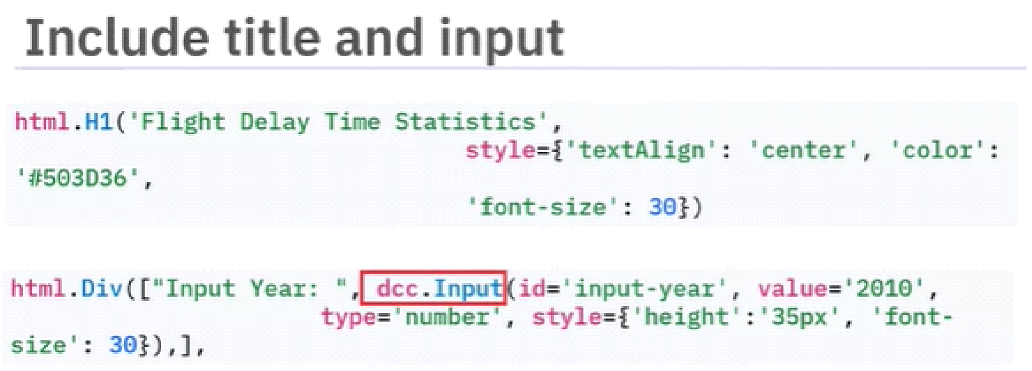
```
@app.callback( Output(component_id='bar-plot', component_property='figure'),
              [ Input(component_id='input-yr', component_property='value'),
                Input(component_id='input-ab', component_property='value')])
def get_graph(entered_year, entered_state):
    # Select data
    df = airline_data[ (airline_data['Year']==int(entered_year)) &
                       (airline_data['OriginState'] == entered_state)]
    # Top 10 airline carrier in terms of number of flights
    ………………………
    fig1.update_layout()
    return fig1
if __name__ == '__main__':
app.run_server(port=8002, host='127.0.0.1', debug=True)
```

Understanding the Lab Environment



Create a skeleton for dash application

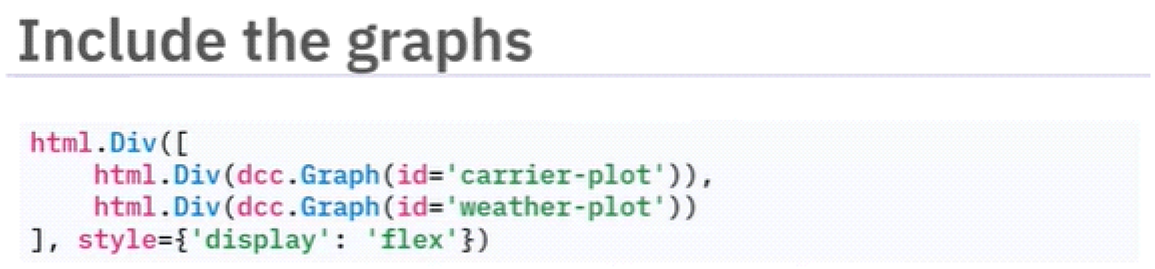
```
# Build dash app layout
app.layout = html.Div(children=[ html.H1(),
                                html.Div(["Input Year: ", dcc.Input()],
                                style={'font-size': 30}),
                                html.Br(),
                                html.Br(),
                                html.Div([
                                        html.Div(),
                                        html.Div()
                                ], style={'display': 'flex'}),

                                html.Div([
                                        html.Div(),
                                        html.Div()
                                ], style={'display': 'flex'}),

                                html.Div(, style={'width':'65%'})
                                ])
```



Include title and input

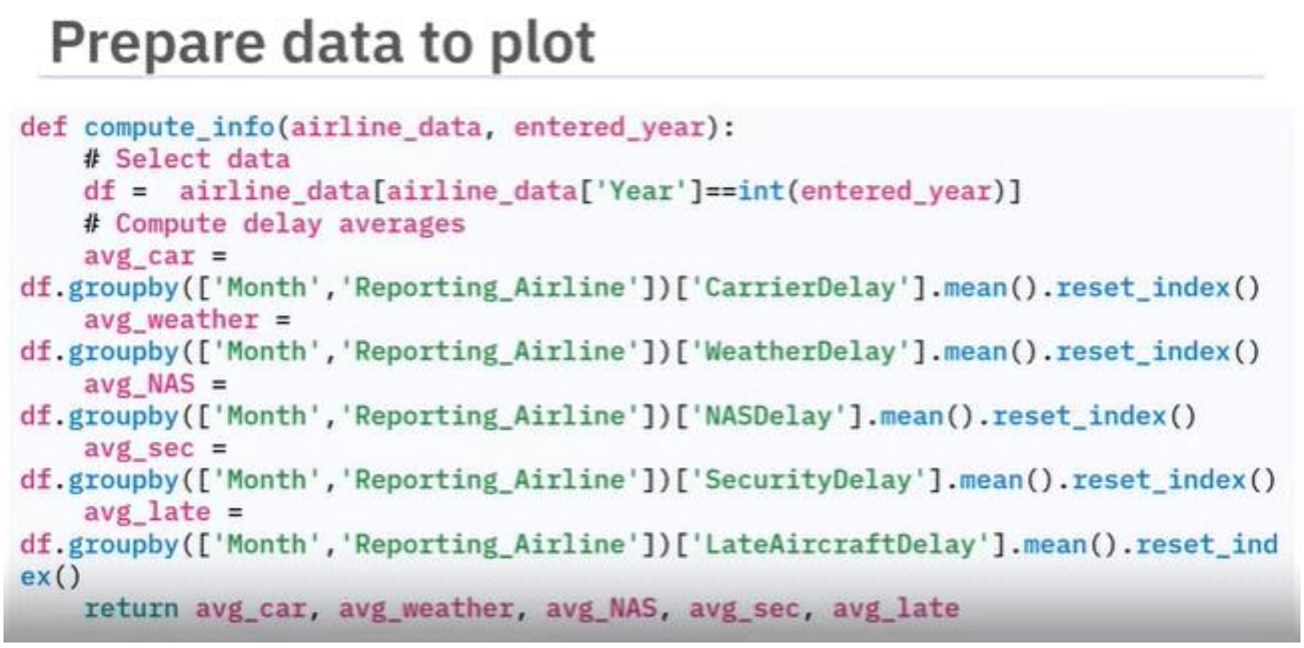
```
html.H1('Flight Delay Time Statistics',
                                style={'textAlign': 'center', 'color':
'#503D36',
                                'font-size': 30})
```

```
html.Div(["Input Year: ", dcc.Input(id='input-year', value='2010',
                       type='number', style={'height':'35px', 'font-
size': 30}),],
```



Include the graphs

```
html.Div([
    html.Div(dcc.Graph(id='carrier-plot')),
    html.Div(dcc.Graph(id='weather-plot'))
], style={'display': 'flex'})
```



Prepare data to plot

```
def compute_info(airline_data, entered_year):
    # Select data
    df =  airline_data[airline_data['Year']==int(entered_year)]
    # Compute delay averages
    avg_car =
df.groupby(['Month','Reporting_Airline'])['CarrierDelay'].mean().reset_index()
    avg_weather =
df.groupby(['Month','Reporting_Airline'])['WeatherDelay'].mean().reset_index()
    avg_NAS =
df.groupby(['Month','Reporting_Airline'])['NASDelay'].mean().reset_index()
    avg_sec =
df.groupby(['Month','Reporting_Airline'])['SecurityDelay'].mean().reset_index()
    avg_late =
df.groupby(['Month','Reporting_Airline'])['LateAircraftDelay'].mean().reset_ind
ex()
    return avg_car, avg_weather, avg_NAS, avg_sec, avg_late
```

屏幕剪辑的捕获时间: 2024/2/2 23:04