Fachhochschule Kiel
University of Applied Sciences

Data Visualization and Visual Analytics
Introduction to Plotly

Study Program Data Science
Prof. Dr. Tillmann Schwörer

Lecture Roadmap

Visualization Types

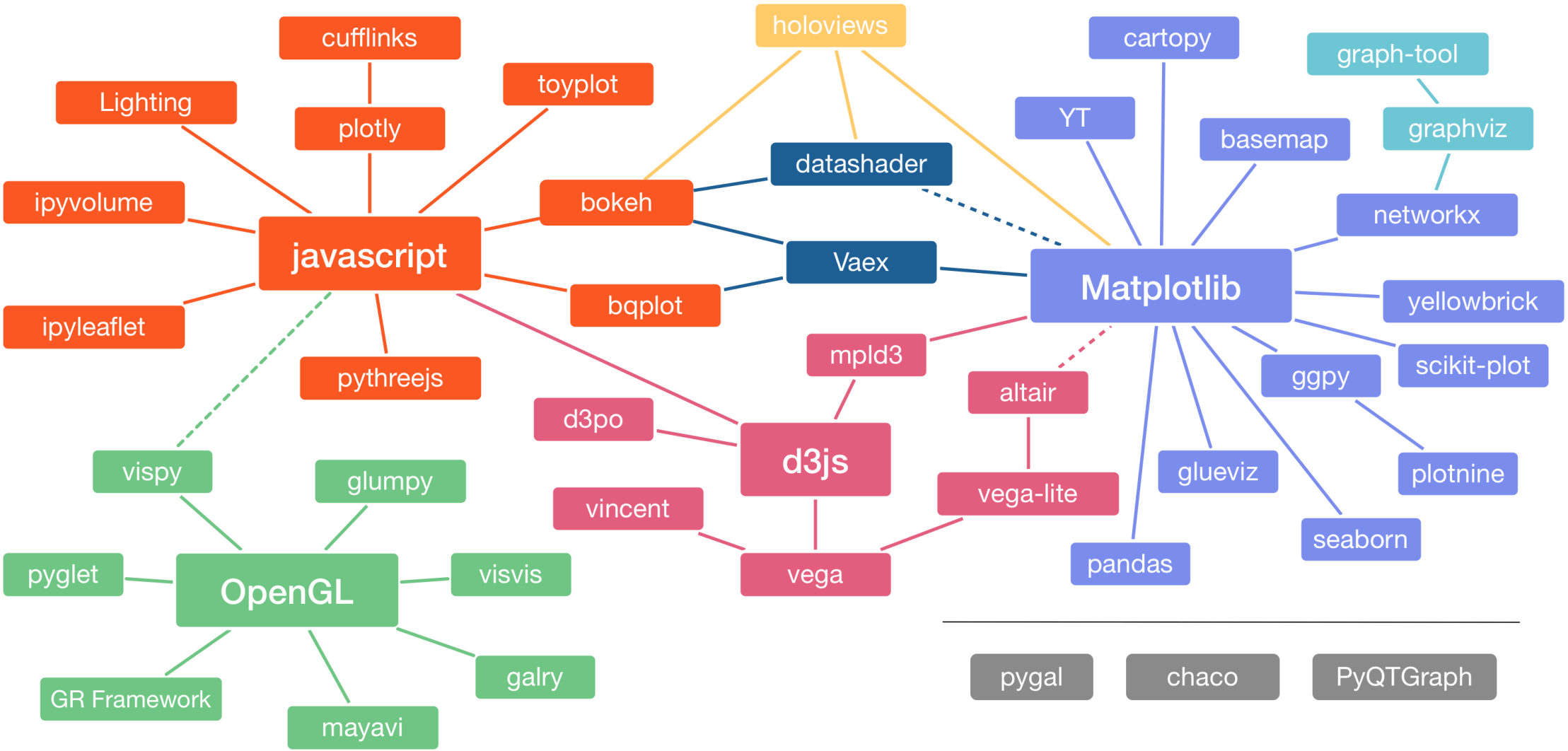
Comparing Categories | Part-to-whole | Relationships | Geospatial | Time | Distributions | Uncertainty | Interactivity | …

Graphic Design Principles

Storytelling

Python Visualization

Python Visualization Landscape



https://pyviz.org/overviews/

What is Plotly

A **JavaScript visualization library** with wrappers in many other languages including Python, R, Matlab and Julia

Advantages:

- Common plots can be created in one line of code (like Seaborn)
- Highly customizable (like Matplotlib)
- Integration with Pandas DataFrames (like Seaborn)
- Interactive features (tooltip, zooming, panning, …)
- Ideal for publishing on websites
- Exportable to vector graphics (svg, pdf) and raster graphics (png, jpg, webP)
- Can be combined with Plotly Dash to integrate plots into web apps
- Open source library with the backing of a company

Plotly Graph Objects vs. Plotly Express

plotly.graph_objects (go):

- Highest level of flexibility and customization

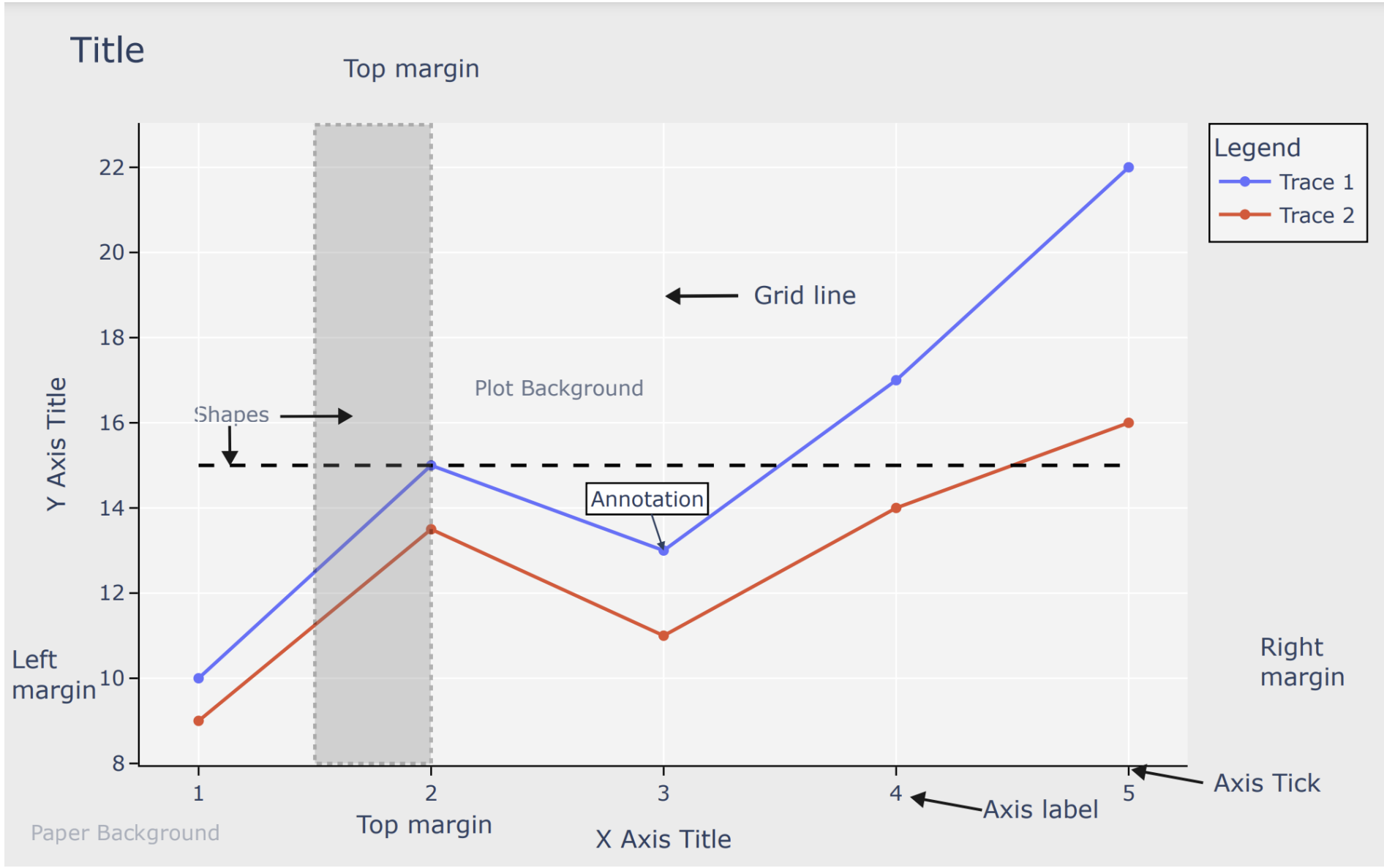
plotly.express (ex)

- Many common plots and fast prototyping
- Accepts Pandas DataFrames
- Automatically handles layout and trace creation for typical scenarios
- Interoperable with the low-level API for further customization

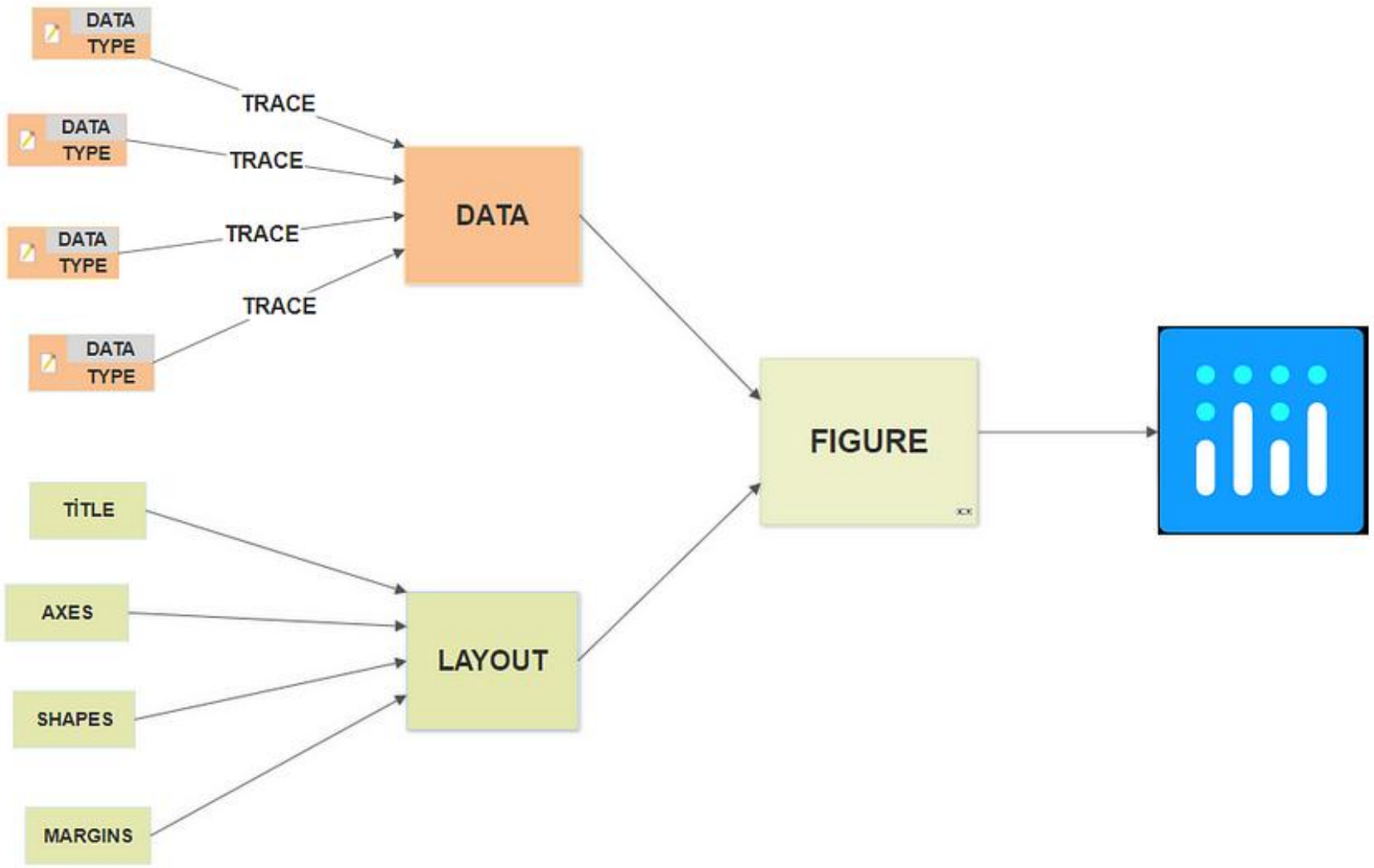
Plotly.figure_factory (ff)

- Non-standard visualizations that cannot be created otherwise

Anatomy of a Plotly Figure



The Plotly Figure



Data and layout

data: a list of traces consisting of

- type: scatter, bar, pie, heatmap, etc.
- data: x values, y values, …
- attributes defining style (marker color, size, …) and interactive behaviour (tooltip)

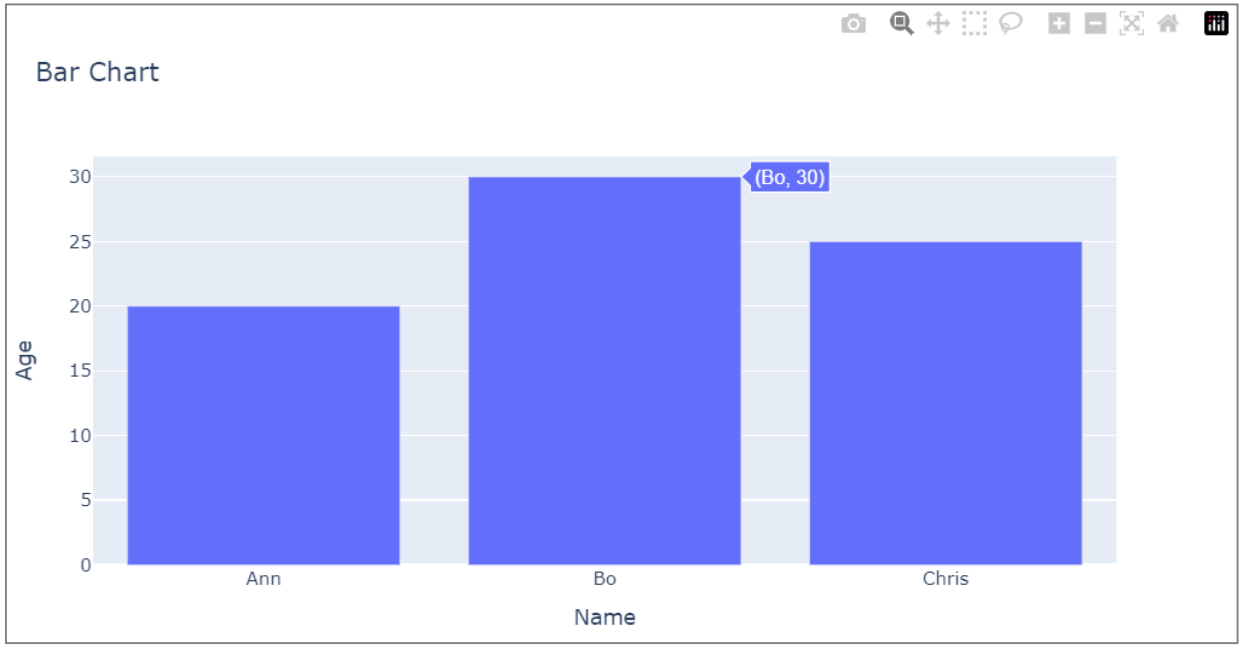
layout: dictionary controlling the non-data components

- Title
- Width and height
- Axes configuration
- …

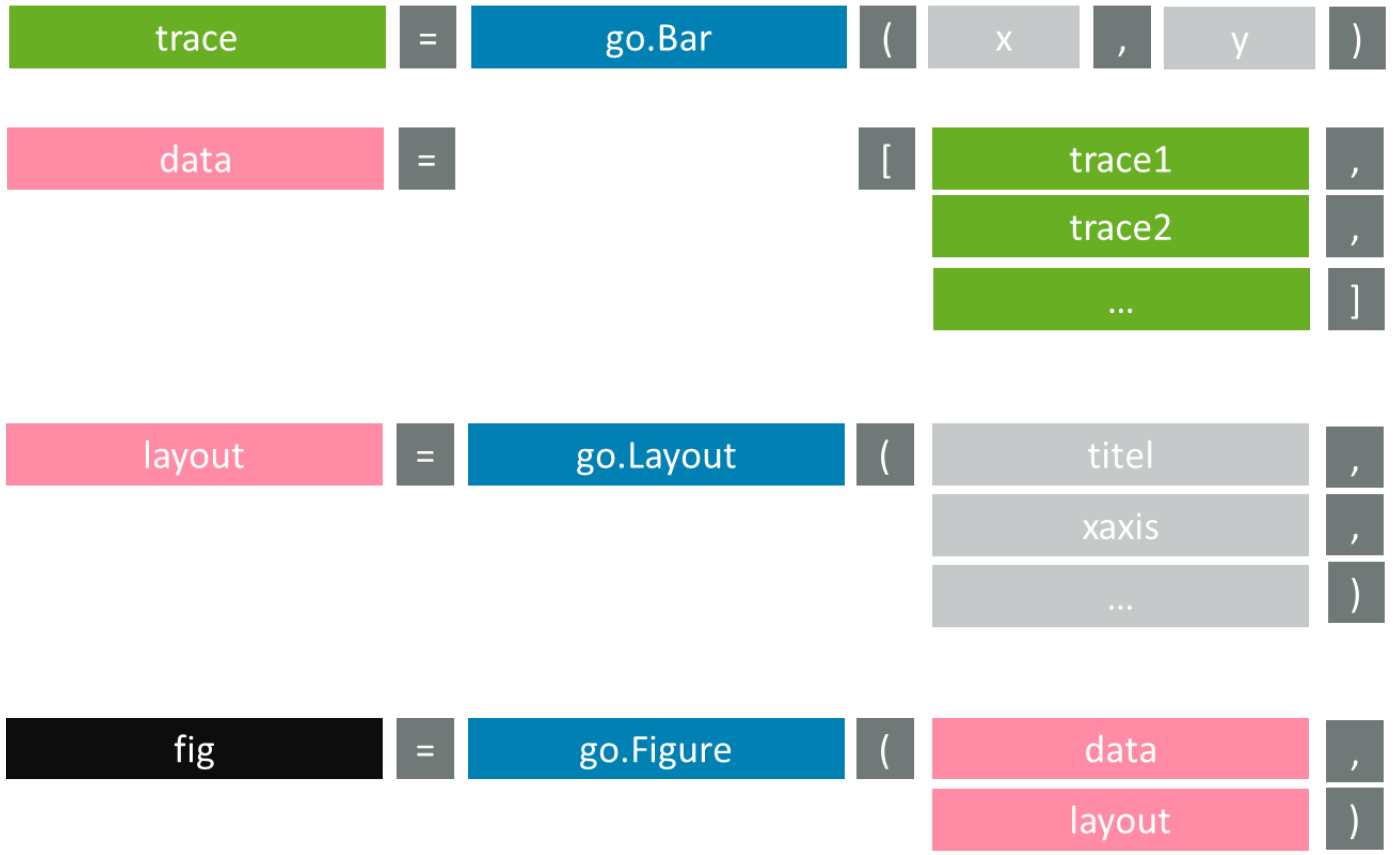
frames : used for animations to transition between different data states

Creating a Bar Chart with Graph Objects

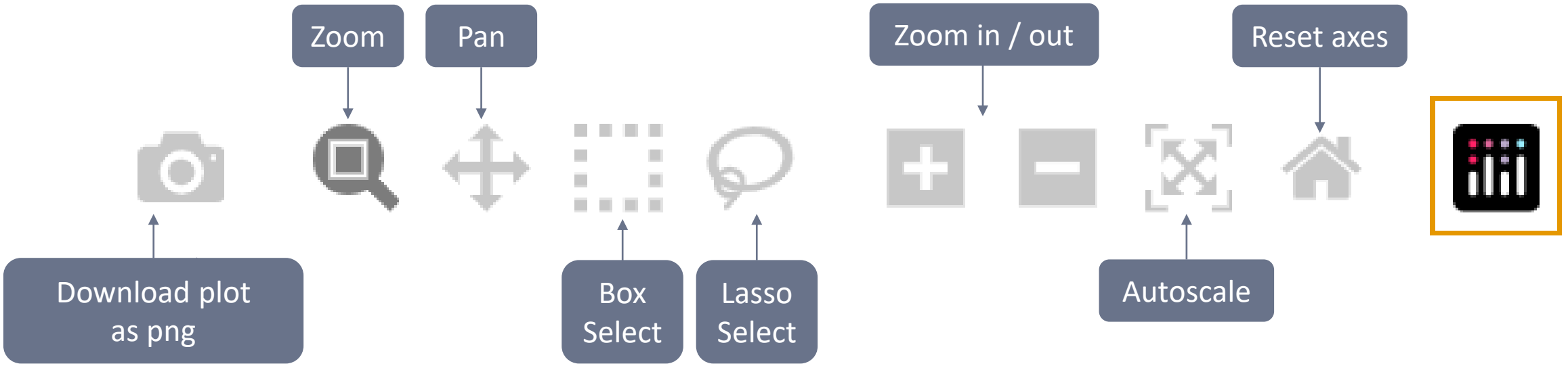
```
trace = go.Bar(x = ['Ann', 'Bo', 'Chris'],
               y = [20, 30, 25])

data = [trace]

layout = go.Layout(title = 'Bar Chart',
                   xaxis = dict(title = 'Name'),
                   yaxis = dict(title = 'Age'))

fig = go.Figure(data = data, layout = layout)
fig.show()
```



Graph Objects Syntax

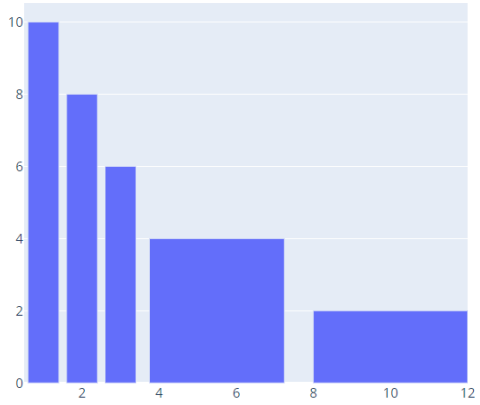


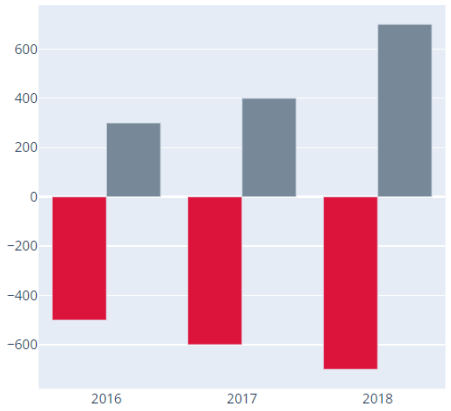
Interactivity

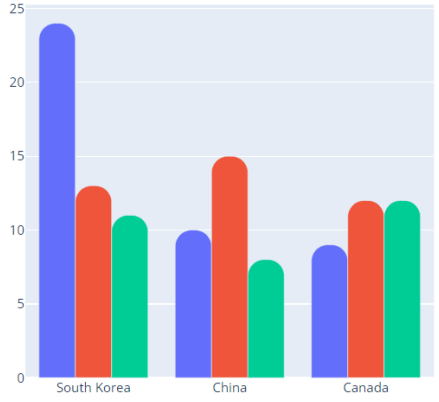


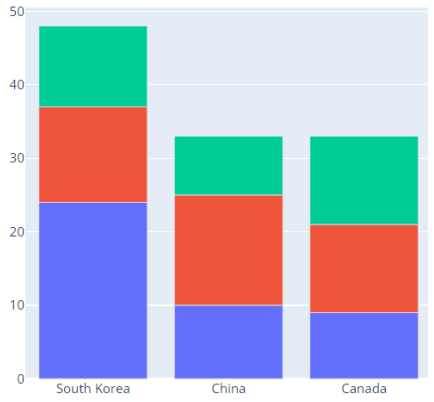
Trace Type: Bar

```
trace = go.Bar( x , y )
```



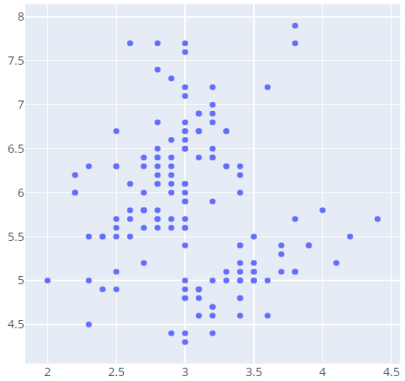
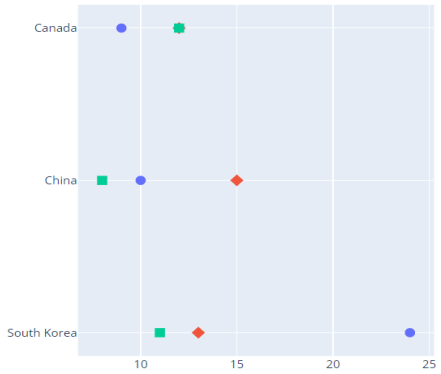


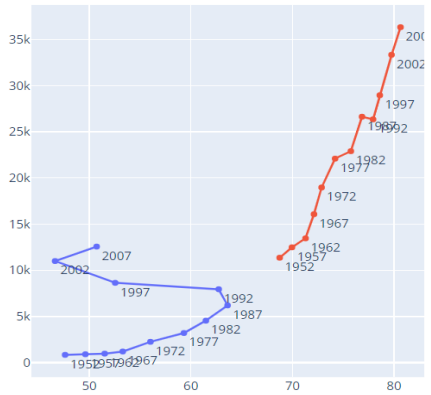


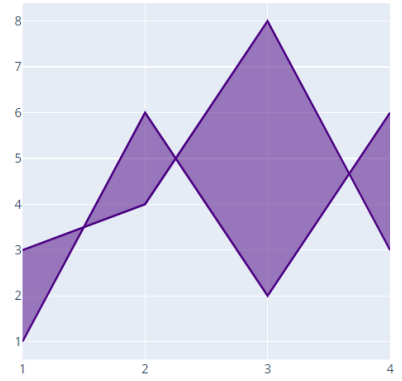
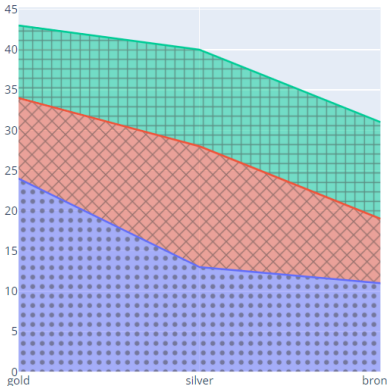


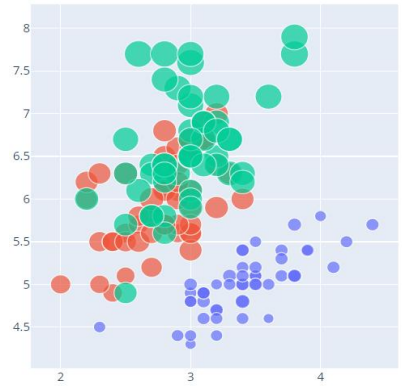
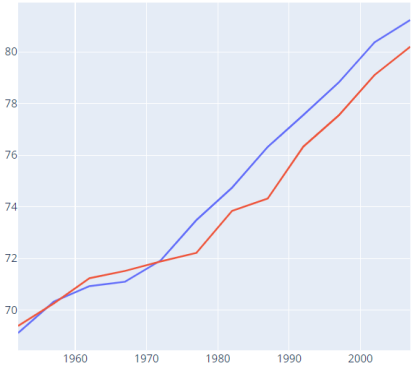
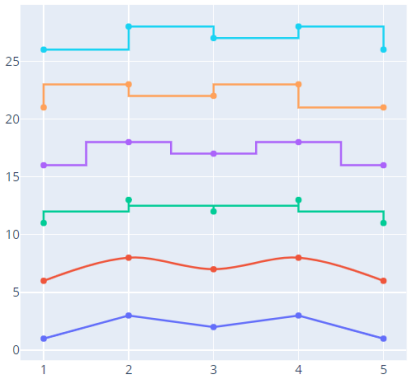
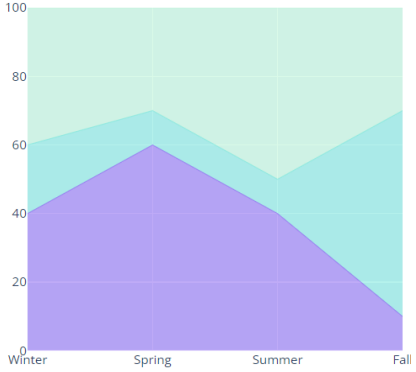
Trace Type: Scatter

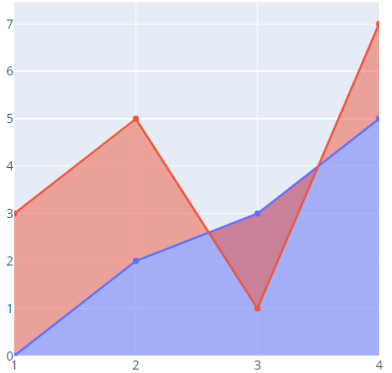
```
trace = go.Scatter( x , y )
```



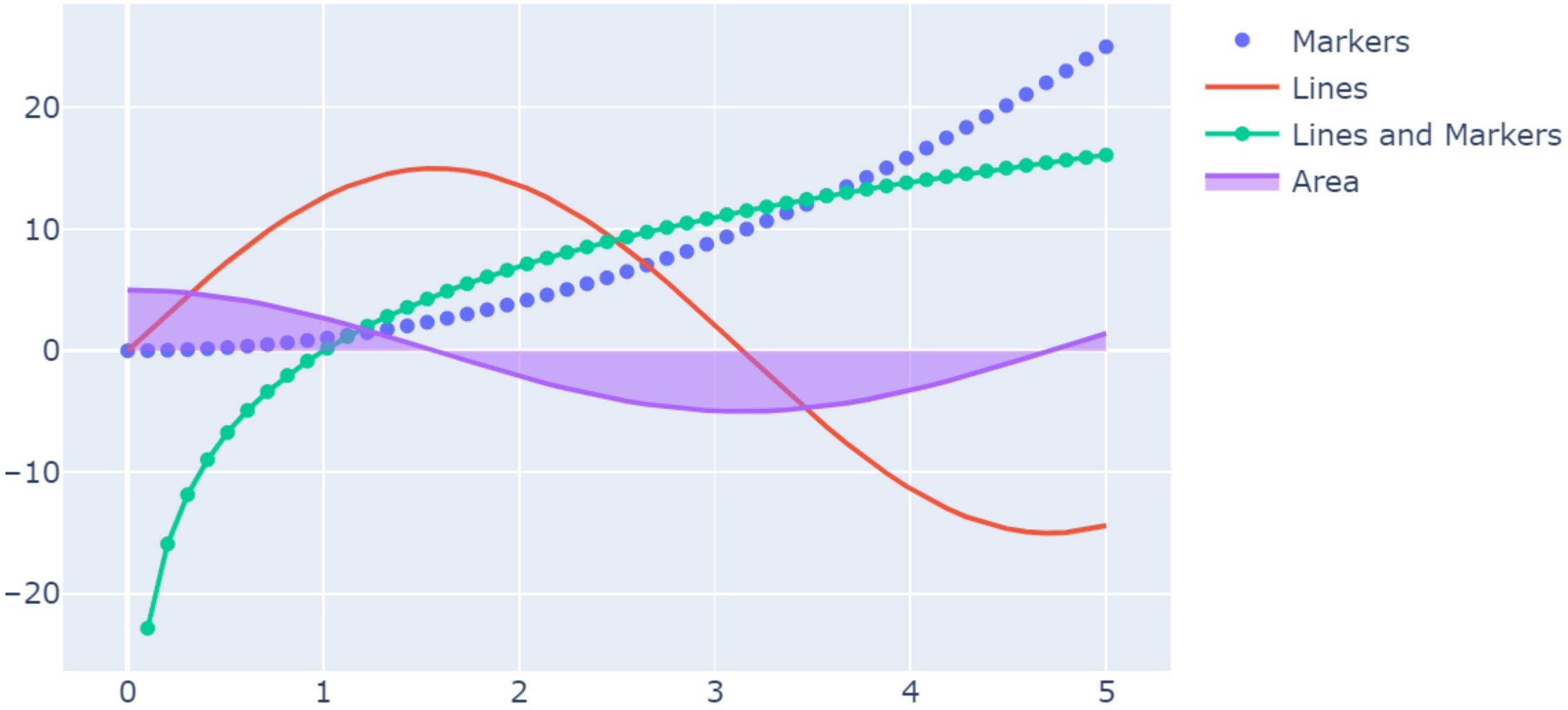


Examples of other trace types

- **Basic Traces**: Heatmap, Pie, Image
- **Distributions**: Box, Violin, Histogram, Contour
- **Cartography**: Choropleth, Choroplethmapbox, Scattermapbox, …
- **Polar Plots**: Barpolar, Scatterpolar, Sunburst
- **3d Plots**: Scatter3d, Mesh3d, Scat

See: https://plotly.com/python/reference/

Multiple Traces



Multiple Traces

```
x = np.linspace(0, 5, 50)
y1 = x**2
y2 = 15*np.sin(x)
y3 = 10*np.log(x)
y4 = 5*np.cos(x)
trace1 = go.Scatter(x=x, y=y1, mode='markers',      name='Markers')
trace2 = go.Scatter(x=x, y=y2, mode='lines',        name='Lines')
trace3 = go.Scatter(x=x, y=y3, mode='lines+markers', name='Lines and Markers')
trace4 = go.Scatter(x=x, y=y4, mode='lines',        name='Area', fill='tozeroy')

go.Figure(data=[trace1, trace2, trace3, trace4])
```

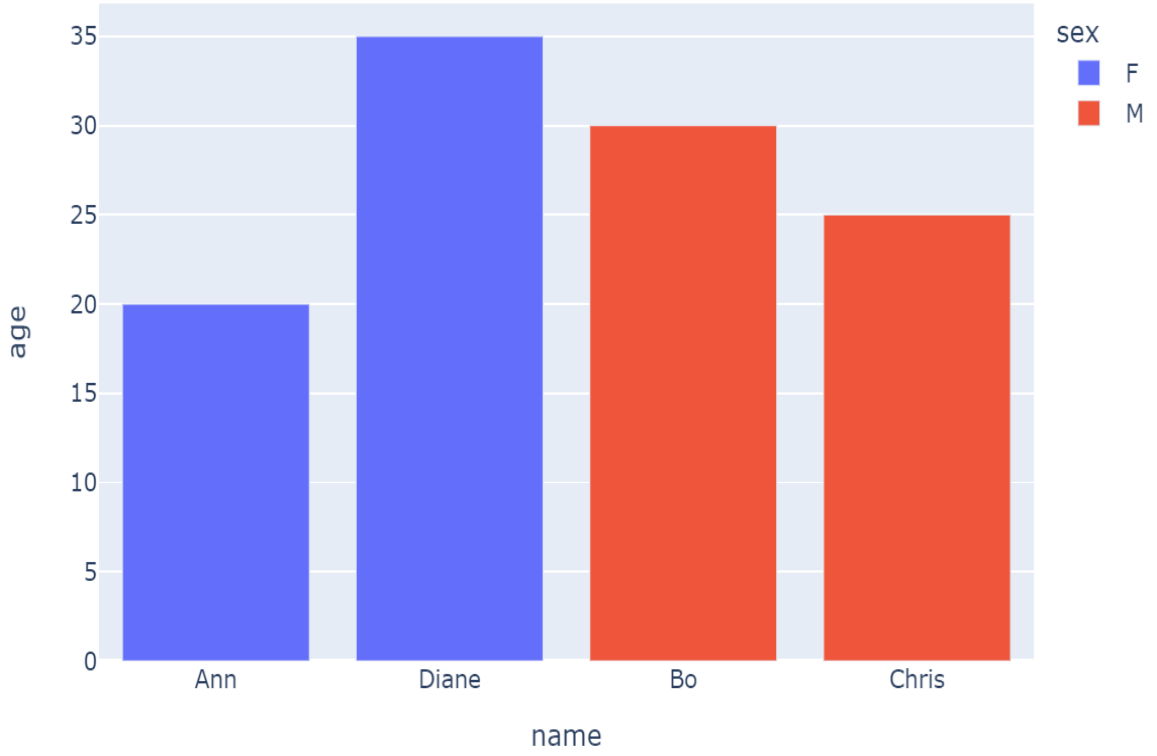
Iteratively adding traces

```
fig = go.Figure()
fig.add_trace(go.Scatter(x=x, y=y1, mode='markers', name='Markers'))
fig.add_trace(go.Scatter(x=x, y=y2, mode='lines', name='Lines'))
```

Or shorter:

```
fig = go.Figure()
fig.add_scatter(x=x, y=y1, mode='markers', name='Markers')
fig.add_scatter(x=x, y=y2, mode='lines', name='Lines')
```

Plotly Express (px)



- Works with DataFrames
- Automatically sets axes titles
- Automatically creates traces or subplots
- Sets useful default values for trace attributes and figure layout

```
df = pd.DataFrame({'name': ['Ann', 'Bo', 'Chris', 'Diane'],
                   'age': [20, 30, 25, 35],
                   'sex': ['F', 'M', 'M', 'F']})

px.bar(df, x='name', y='age', color='sex')
```

Layout Customization

We can customize figures either

- At figure creation, *if an argument exists*
- After figure creation using the update_layout() method

```
fig.update_layout({'title':{'text':'A New Title'}})
```

Plotly Documentation

- **Main documentation**: https://plotly.com/python/
- **Graph objects documentation**: https://plotly.com/python-api-reference/plotly.graph_objects.html
- **go.Figure documentation**: https://plotly.com/python-api-reference/generated/plotly.graph_objects.Figure.html