Introduction to widgets

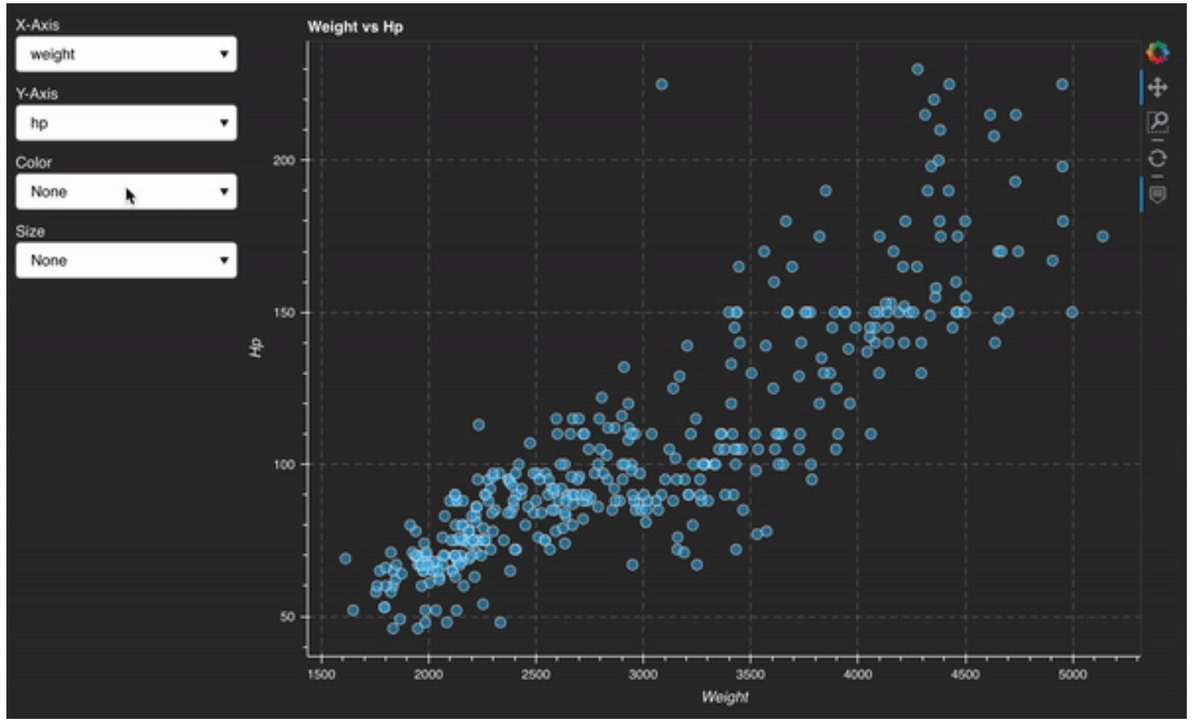
INTERACTIVE DATA VISUALIZATION WITH BOKEH

George Boorman
Core Curriculum Manager, DataCamp

datacamp

Widgets

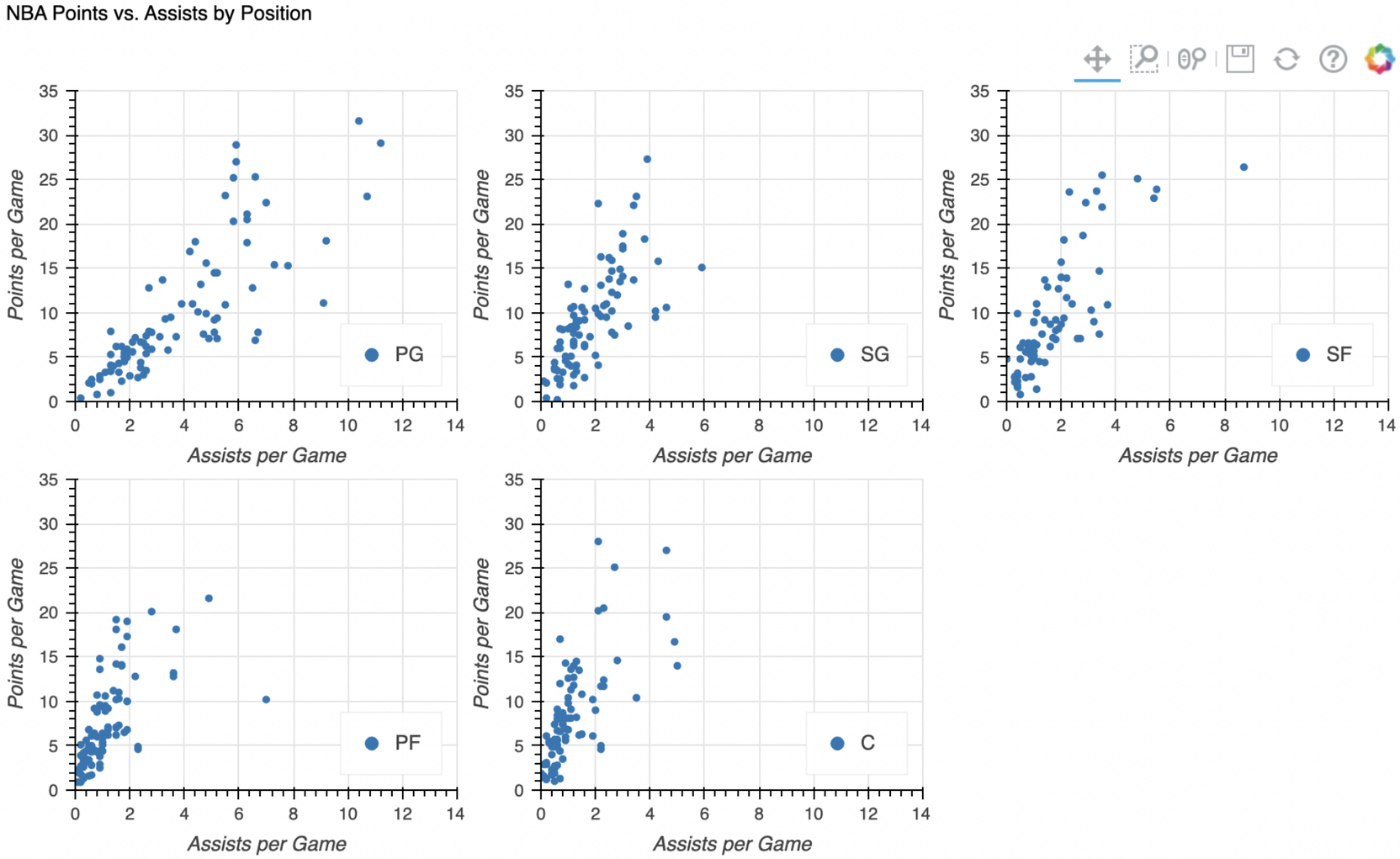
- 8 Categories
  - `buttons`
  - `groups`
  - `icons`
  - `inputs`
  - `markups`
  - `sliders`
  - `tables`
  - `widgets`



Layout and Div

```
from bokeh.models import Div
from bokeh.layouts import layout, gridplot
pg = nba.loc[nba["position"] == "PG"]
sg = nba.loc[nba["position"] == "SG"]
sf = nba.loc[nba["position"] == "SF"]
pf = nba.loc[nba["position"] == "PF"]
c = nba.loc[nba["position"] == "C"]
title = Div(text="NBA Points vs. Assists by Position")
plots = []
for df in [pg, sg, sf, pf, c]:
    fig = figure(x_axis_label="Assists per Game", y_axis_label="Points per Game",
                 x_range=(0,14), y_range=(0,35))
    fig.circle(x=df["assists"], y=df["points"], legend_label=df["position"].unique()[0])
    fig.legend.location = "bottom_right"
    plots.append(fig)
output_file(filename="subplots_with_Div.html")
show(layout([title], [gridplot(plots, ncols=3)]))
```

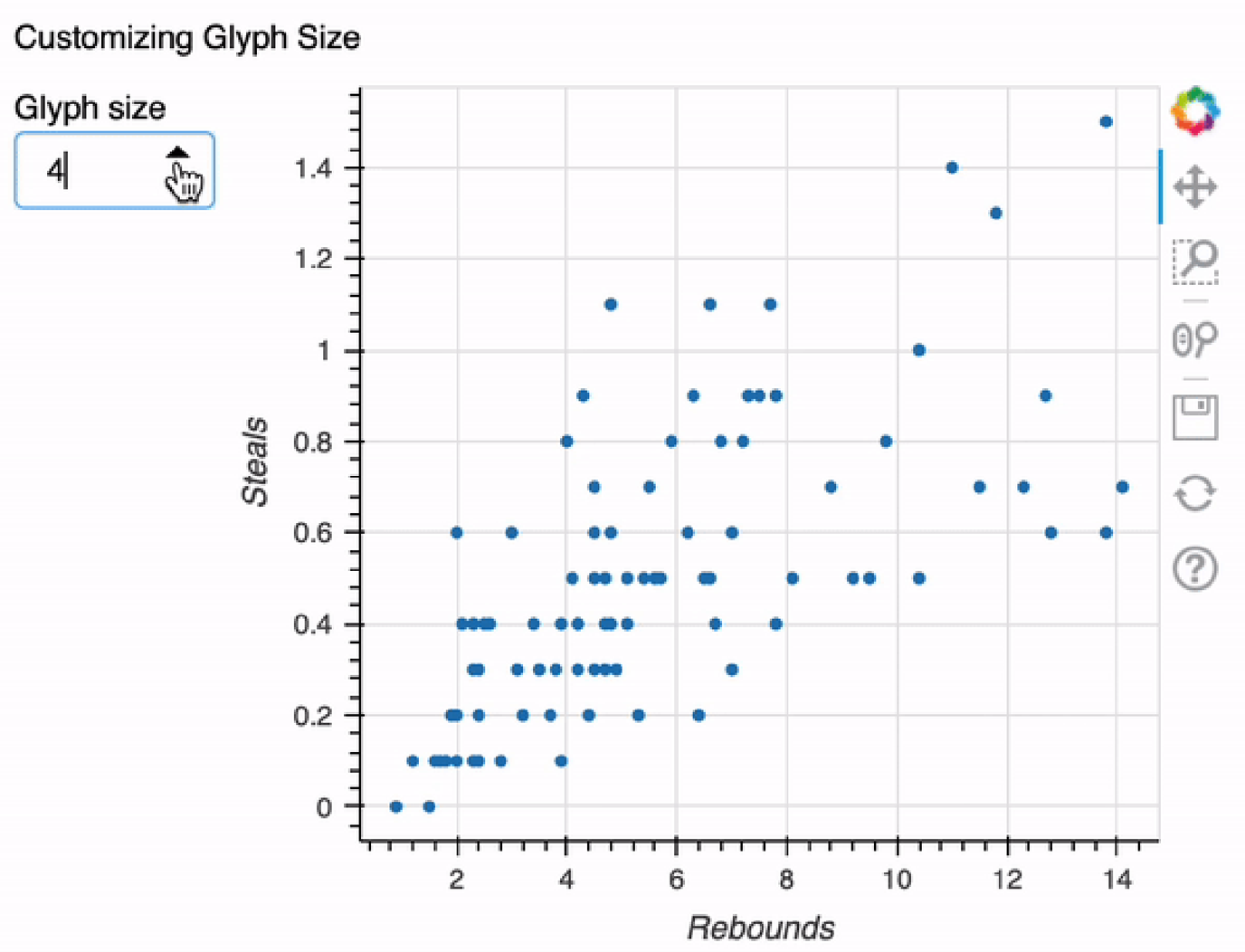
Gridplot with title



Building a spinner widget

```
from bokeh.models import Spinner
fig = figure(x_axis_label="Rebounds", y_axis_label="Steals")
scatter = fig.circle(x="rebounds", y="steals", source=source)
spinner = Spinner(title="Glyph size", low=1, high=40, step=0.5, value=4, width=80)
spinner.js_link("value", scatter.glyph, "size")
show(layout([title], [spinner, fig]))
```

Spin into action



The dataset

```
print(stocks.head())
```

```
     date          open      high      low       close     volume      name
0    2014-06-02    90.5656   90.6899   88.9285   89.8071   92337903    AAPL
1    2014-06-03    89.7799   91.2485   89.7499   91.0771   73231620    AAPL
2    2014-06-04    91.0628   92.5556   90.8728   92.1171   83870521    AAPL
3    2014-06-05    92.3142   92.7670   91.8013   92.4785   75951141    AAPL
4    2014-06-06    92.8428   93.0370   92.0671   92.2242   87620911    AAPL
```

Let's practice!

INTERACTIVE DATA VISUALIZATION WITH BOKEH

Slider widgets

INTERACTIVE DATA VISUALIZATION WITH BOKEH

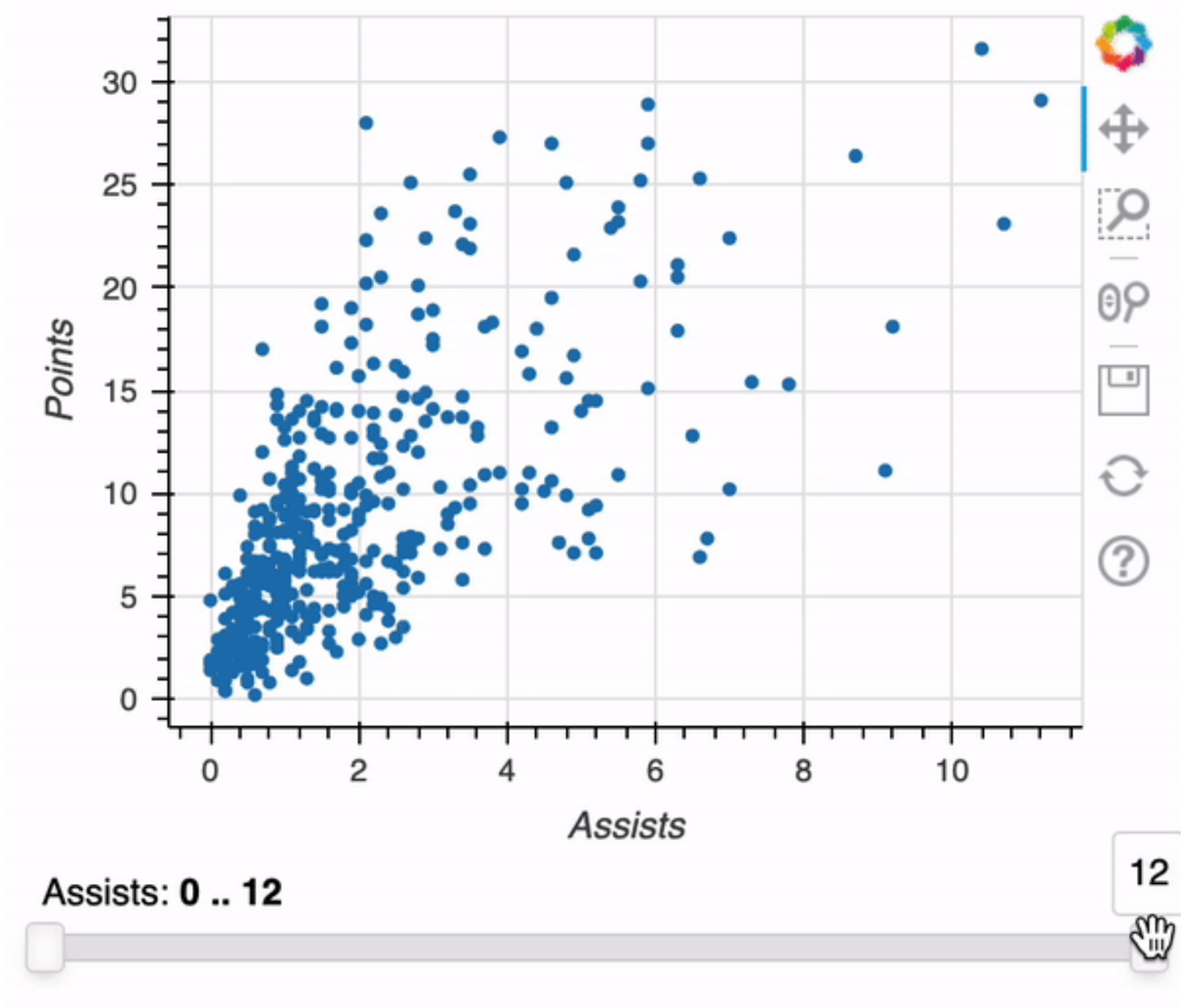
George Boorman
Core Curriculum Manager, DataCamp

datacamp

RangeSlider widget

```
from bokeh.models import RangeSlider
fig = figure(x_axis_label="Assists", y_axis_label="Points")
fig.circle(x="assists", y="points", source=source)
x_slider = RangeSlider(title="Assists", start=0, end=12, value=(0,12), step=0.5)
x_slider.js_link("value", fig.x_range, "start", attr_selector=0)
x_slider.js_link("value", fig.x_range, "end", attr_selector=1)
output_file(filename="x_axis_slider.html")
show(layout([fig, x_slider]))
```

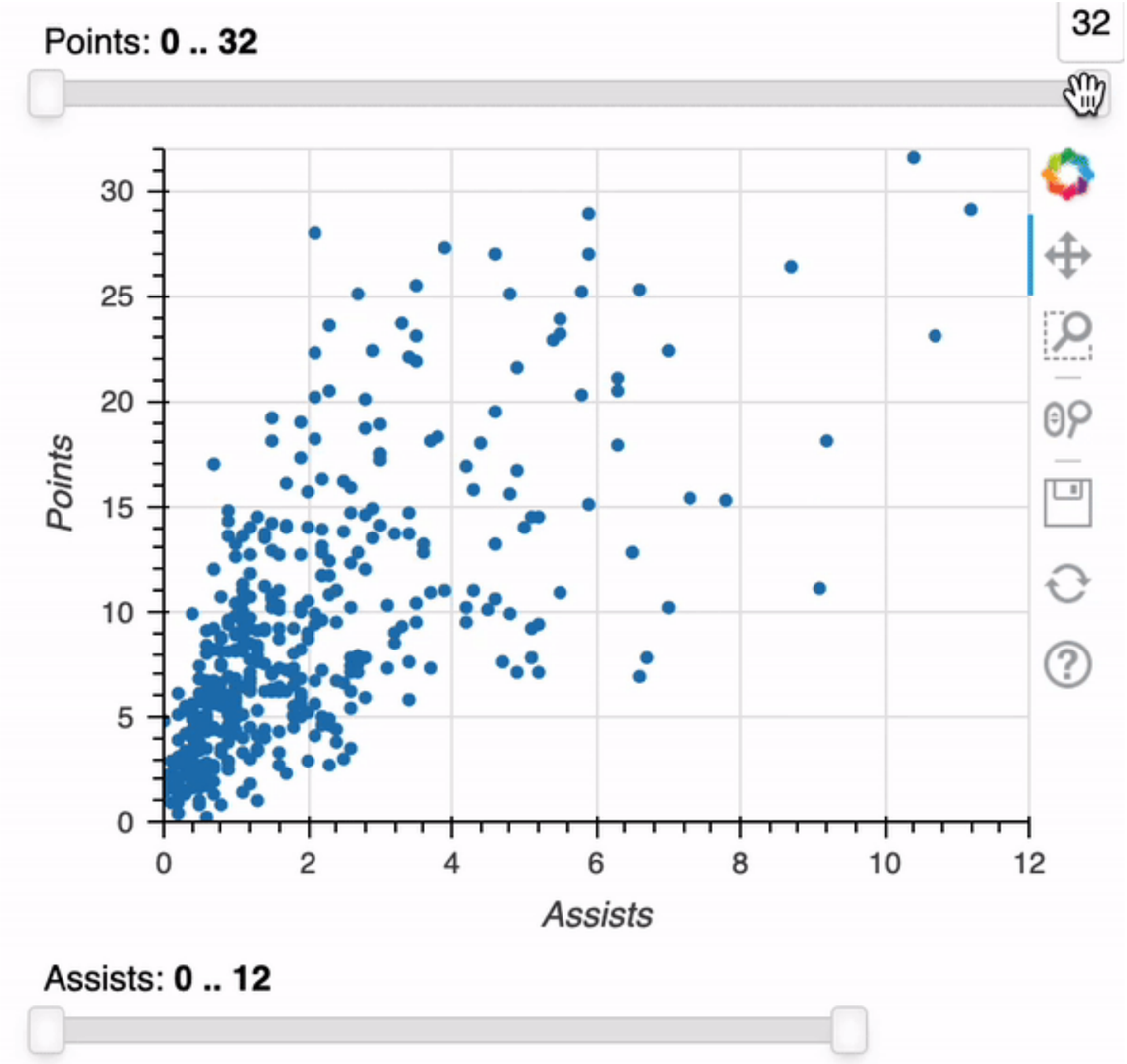
RangeSlider



Multiple widgets

```
fig = figure(x_axis_label="Assists", y_axis_label="Points", height=300, width=400)
fig.circle(x="assists", y="points", source=source)
x_slider = RangeSlider(title='Assists', start=0, end=12, value=(0,12), step=0.5)
x_slider.js_link("value", fig.x_range, "start", attr_selector=0)
x_slider.js_link("value", fig.x_range, "end", attr_selector=1)
y_slider = RangeSlider(title='Points', start=0, end=32, value=(0,32), step=1)
y_slider.js_link("value", fig.y_range, "start", attr_selector=0)
y_slider.js_link("value", fig.y_range, "end", attr_selector=1)
output_file(filename="multiple_RangeSliders.html")
show(layout(column(y_slider, fig), [x_slider]))
```

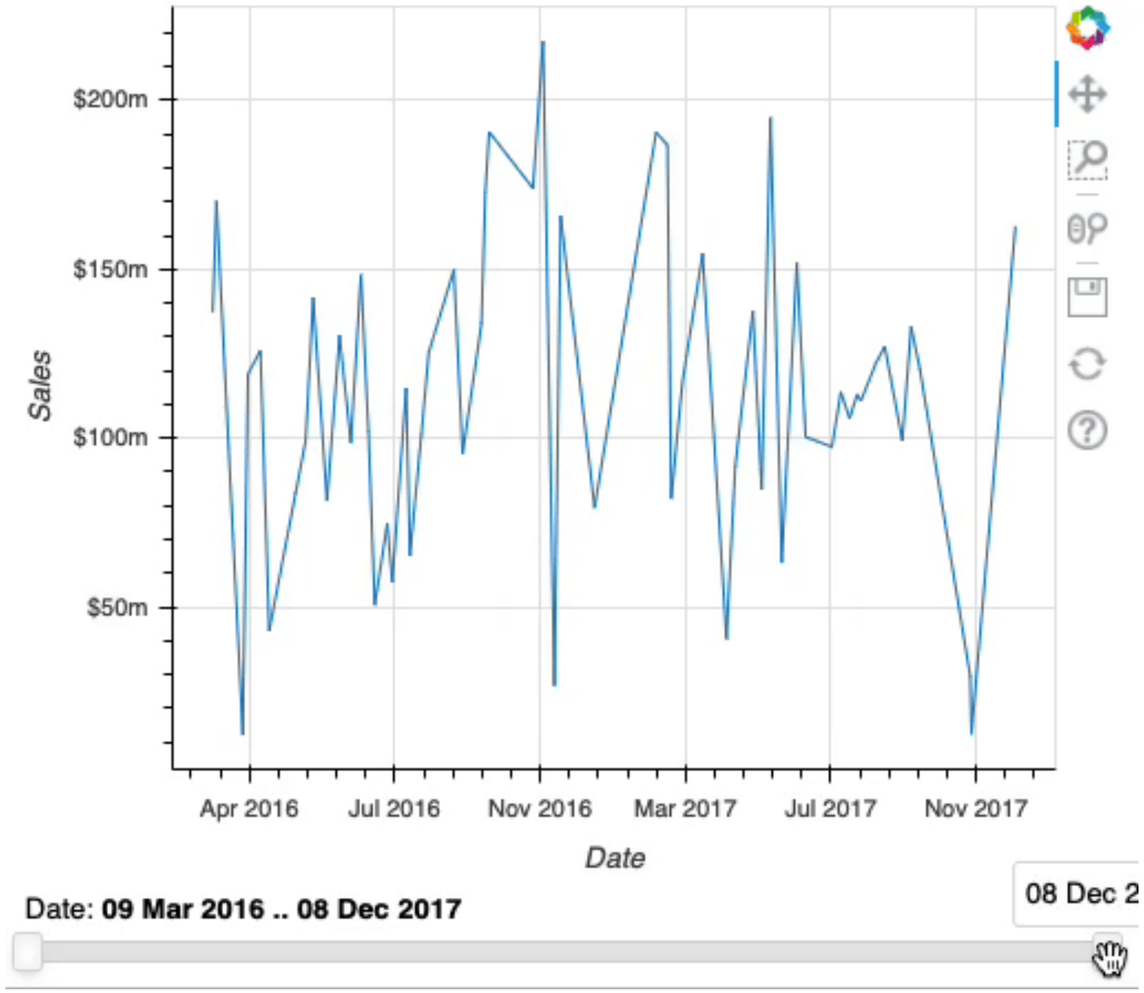
RangeSlider for both axes



DateRangeSlider widget

```
from bokeh.models import DateRangeSlider
melb_sales = melb.groupby('date', as_index=False)['price'].sum()
source = ColumnDataSource(data=melb_sales)
fig = figure(x_axis_label="Date", y_axis_label="Sales")
fig.line(x="date", y="price", source=source)
fig.xaxis[0].formatter = DatetimeTickFormatter(months="%b %Y")
fig.yaxis[0].formatter = NumeralTickFormatter(format="$0a")
slider = DateRangeSlider(title="Date", start=melb_sales["date"].min(),
                                 end=melb_sales["date"].max(),
                                 value=("2016, 3, 9", "2017, 12, 8"), step=1)
slider.js_link("value", fig.x_range, "start", attr_selector=0)
slider.js_link("value", fig.x_range, "end", attr_selector=1)
output_file(filename="melbourne_sales.html")
show(layout([fig, date_slider]))
```

DateRangeSlider



Let's practice!

INTERACTIVE DATA VISUALIZATION WITH BOKEH

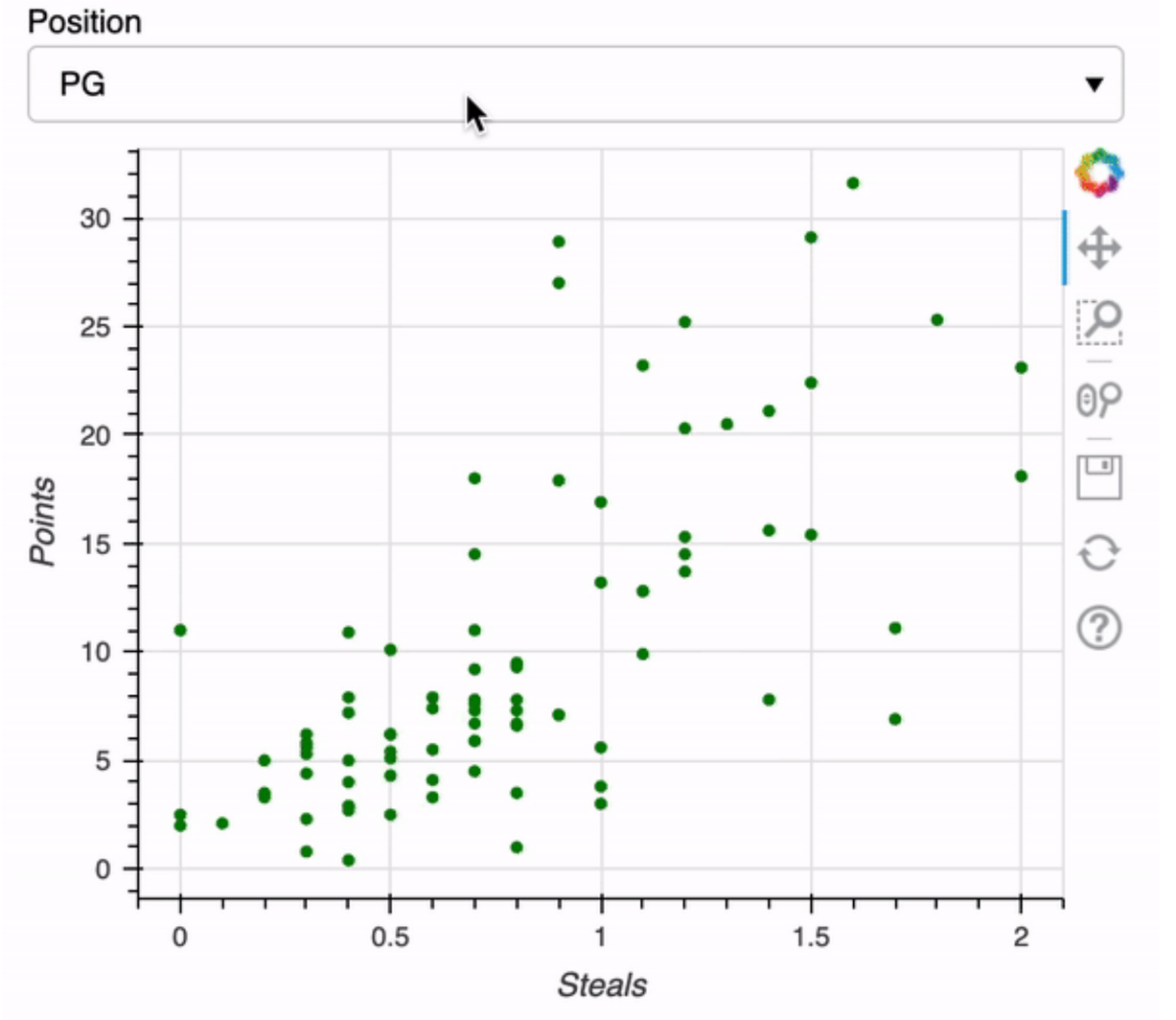
Select widgets

INTERACTIVE DATA VISUALIZATION WITH BOKEH

George Boorman
Core Curriculum Manager, DataCamp

datacamp

Select widget



JavaScript

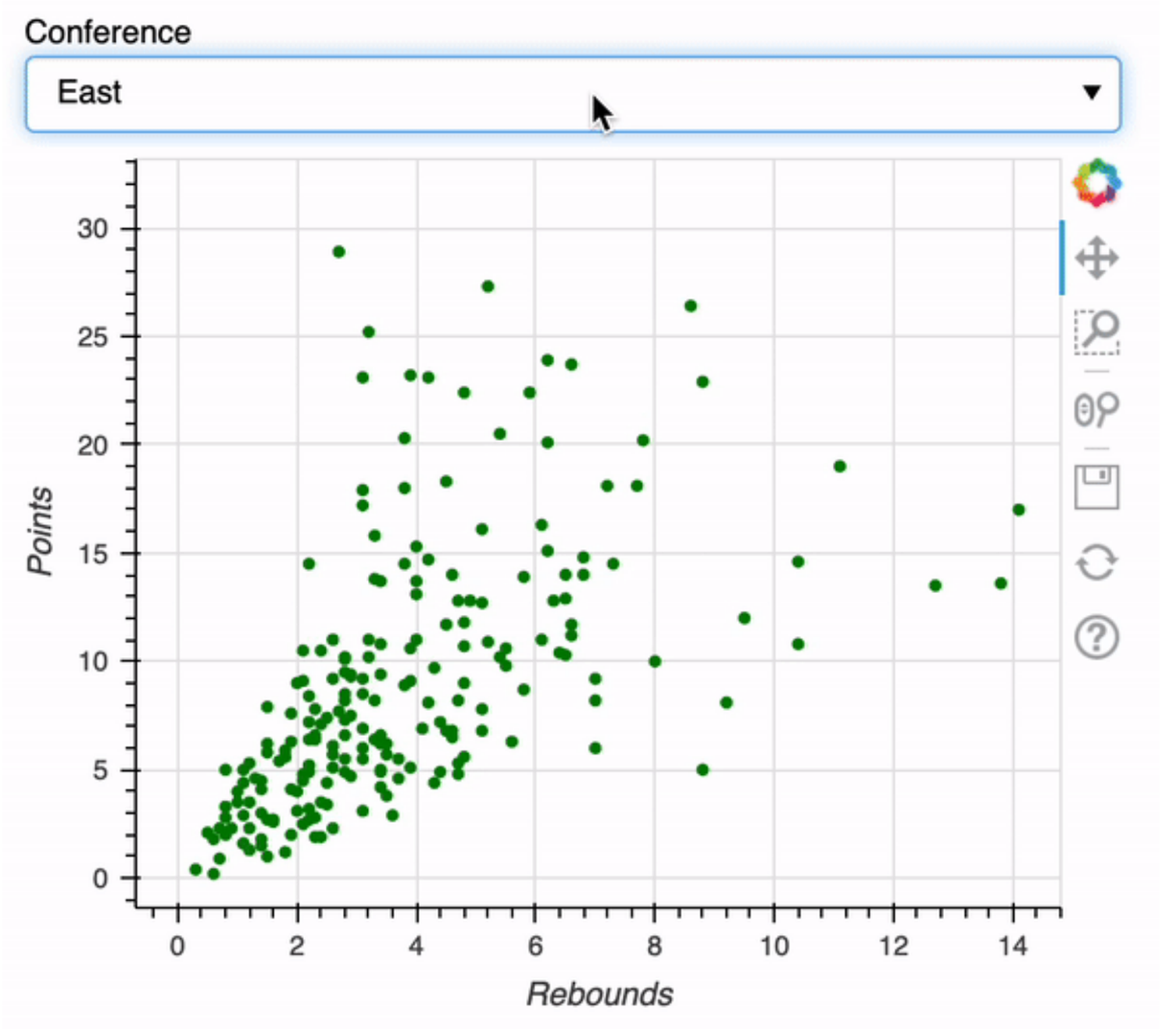
Setting up the Select widget

```
from bokeh.models import Select, CustomJS
fig = figure(x_axis_label="Rebounds", y_axis_label="Points")
east_glyph = fig.circle(x="rebounds", y="points", color="green", source=east)
west_glyph = fig.circle(x="rebounds", y="points", color="red", source=west)
west_glyph.visible = False
menu = ["East", "West"]
```

Building an interactive function

```
callback = CustomJS(args=dict(scatter_1=east_glyph, scatter_2=west_glyph),
                    code="""scatter_1.visible = true
                            scatter_2.visible = true
                            if (this.value == "East") {scatter_2.visible = false}
                            else {scatter_1.visible = false}""")
menu = Select(options=menu, value="East", title="Conference")
menu.js_on_change("value", callback)
output_file(filename="select_widget.html")
layout=column(menu, fig)
show(layout)
```

The result!


Conference
East
Points
Rebounds
0
5
10
15
20
25
30
0
2
4
6
8
10
12
14

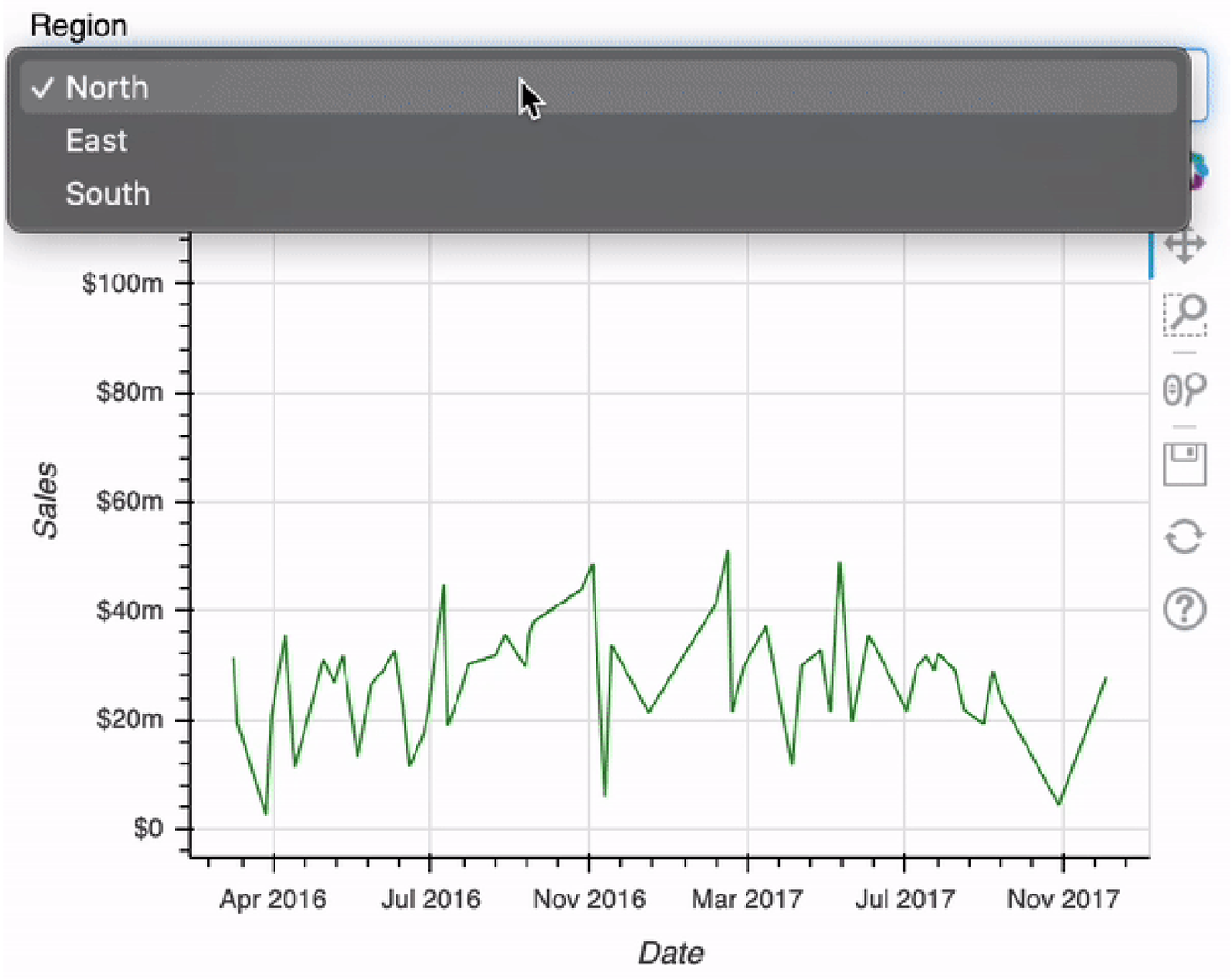
Selecting from three options

```
fig = figure(x_axis_label="Date", y_axis_label="Sales")
north_glyphs = fig.line(x="date", y="price", color="green", source=north)
east_glyphs = fig.line(x="date", y="price", color="red", source=east)
south_glyphs = fig.line(x="date", y="price", color="purple", source=south)
fig.xaxis[0].formatter = DatetimeTickFormatter(months="%b %Y")
fig.yaxis[0].formatter = NumeralTickFormatter(format="$0a")
east_glyphs.visible = False
south_glyphs.visible = False
menu = Select(options=["North", "East", "South"], value="North", title="Region")
```

Building three interactions

```
callback = CustomJS(args=dict(line_1=north_glyphs, line_2=east_glyphs,
                              line_3=south_glyphs), code="""
                              line_1.visible = true
                              line_2.visible = true
                              line_3.visible = true
                              if (this.value == "North") {line_2.visible = false
                                  line_3.visible = false} else {line_1.visible = false}
                              if (this.value == "East") {line_1.visible = false
                                  line_3.visible = false} else {line_2.visible = false}
                              if (this.value == "South") {line_1.visible = false
                                  line_2.visible = false} else {line_3.visible = false}""")
menu.js_on_change("value", callback)
output_file(filename="melbourne_regions_widget.html")
layout=column(menu, fig)
show(layout)
```

Select widget with line plots



Using Select for different figures

```
stocks["market_cap"] = stocks["volume"] * stocks["close"]
ebay = stocks.loc[stocks["name"] == "EBAY"]
source = ColumnDataSource(data=ebay)
fig = figure(x_axis_label="Date", y_axis_label="Stock Price")
fig_two = figure(x_axis_label="Date", y_axis_label="Market Cap")
fig_three = figure(x_axis_label="Date", y_axis_label="Volume")
price = fig.line(x="date", y="close", color="green", source=source)
market_cap = fig_two.line(x="date", y="market_cap", color="red", source=source)
volume = fig_three.line(x="date", y="volume", color="purple", source=source)
fig_two.visible = False
fig_three.visible = False
market_cap.visible = False
volume.visible = False
```

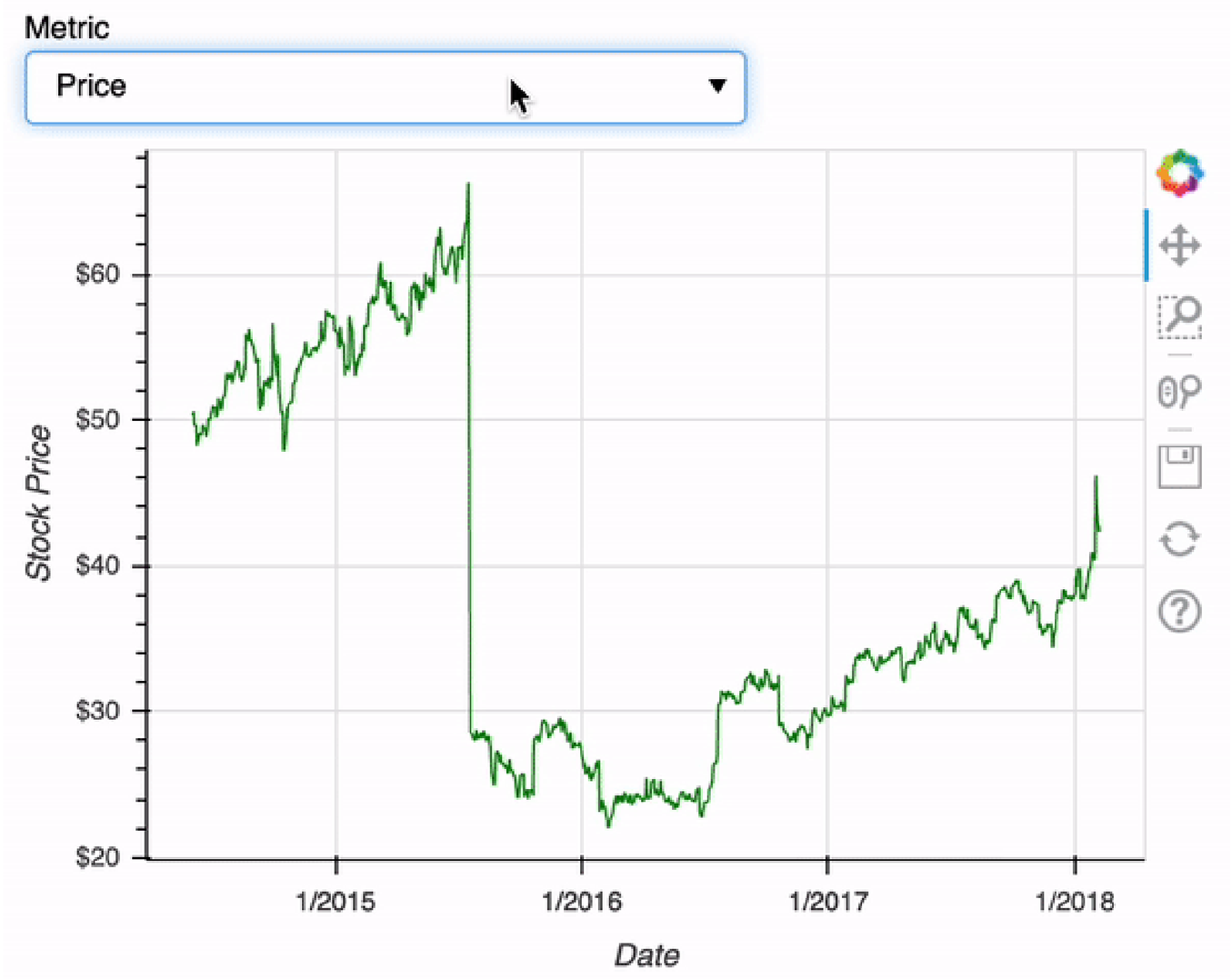
Building figure interaction

```
menu = Select(options=["Price", "Market Cap", "Volume"], value="Price", title="Metric")
callback = CustomJS(args=dict(plot_one=fig, plot_two=fig_two, plot_three=fig_three,
                              line_1=price, line_2=market_cap, line_3=volume),
                code="""
plot_one.visible = true
plot_two.visible = true
plot_three.visible = true
line_1.visible = true
line_2.visible = true
line_3.visible = true
if (this.value == "Price") {plot_two.visible = false
                            plot_three.visible = false
                            line_2.visible = false
                            line_3.visible = false}
    else {plot_one.visible = false
          line_1.visible = false}
```

Completing the callback

```
"""
if (this.value == "Market Cap") {plot_one.visible = false
                                 plot_three.visible = false
                                 line_1.visible = false
                                 line_3.visible = false}
    else {plot_two.visible = false
          line_2.visible = false}
if (this.value == "Volume") {plot_one.visible = false
                             plot_two.visible = false
                             line_1.visible = false
                             line_2.visible = false}
    else {plot_three.visible = false
          line_3.visible = false}
""")
menu.js_on_change("value", callback)
output_file(filename="multiple_figures,html")
layout=layout([menu], [fig, fig_two, fig_three])
show(layout)
```

Switching figures



Let's practice!

INTERACTIVE DATA VISUALIZATION WITH BOKEH

Congratulations!

INTERACTIVE DATA VISUALIZATION WITH BOKEH

George Boorman
Core Curriculum Manager, DataCamp

Recap

- Chapter 1: Introduction to Bokeh
- Chapter 2: Customizing visualizations
- Chapter 3: Storytelling with visualizations
- Chapter 4: Using widgets

What now?

- Reshaping Data with Pandas
- Working with Categorical Data in Python
- Time Series with Python

Thank you!

INTERACTIVE DATA VISUALIZATION WITH BOKEH