Altair/Vega:
интерактивные графики из ноутбука на сайт
или d3.js на халяву

Илья Тимофеев, Старший партнер, ТТС Консалтинг

Supplementary materials (demo, notebook): https://iliatimofeev.github.io/PyData2019Altair/

Who am I ?

Ilia Timofeev, Senior partner, TTS Consulting

- I draw slides with charts on a daily basis since … I remember myself
- I'm Altair User since 2017 and a bit contributor
- An author of GpdVega which integrates Altair and GeoPandas
- Also Vega/Vega-Lite tester

Supplementary materials (demo, notebook): https://iliatimofeev.github.io/PyData2019Altair/

Altair: Python API for Vega-Lite Grammar of Graphics

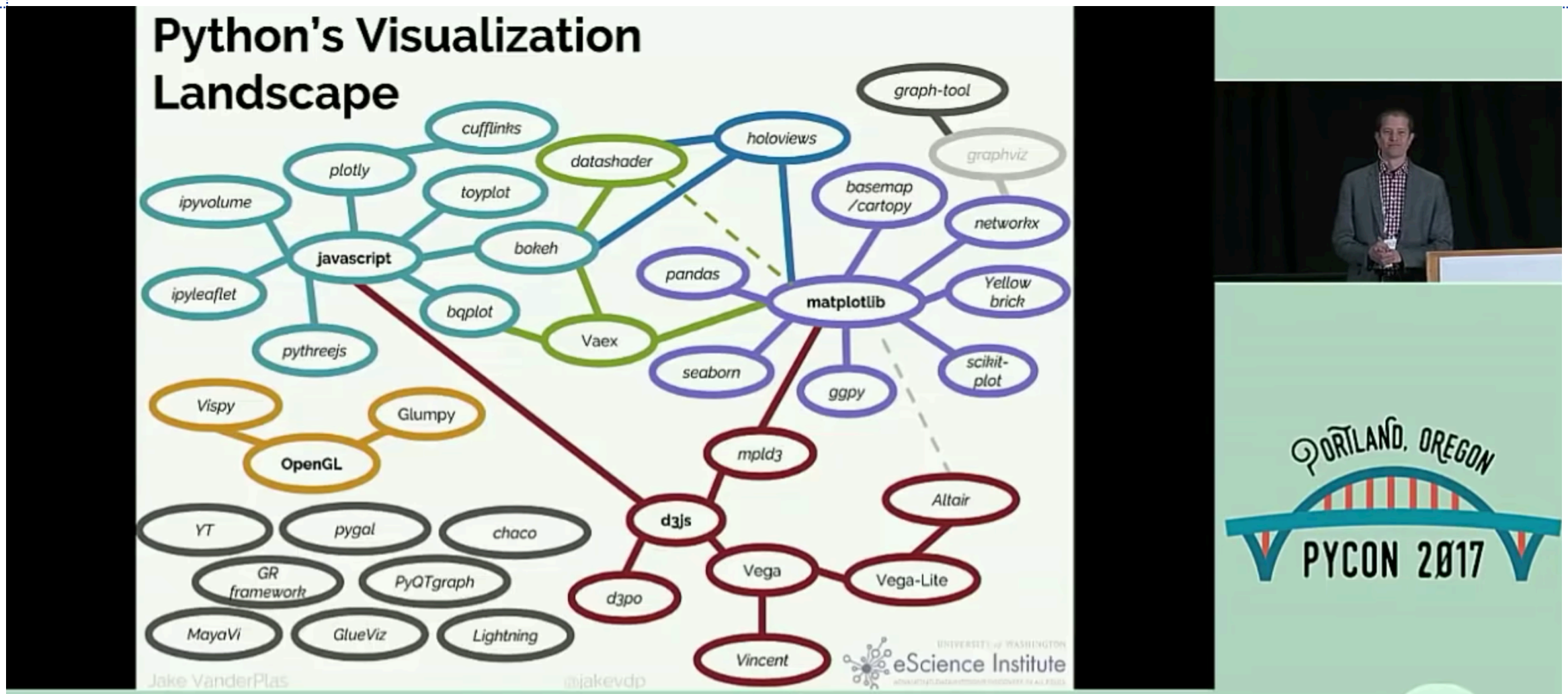
Agenda

- Why Altair ?
  - Stack of Altair
  - Motivation
  - An example
- API for grammar of interactive graphics
  - Grammar of Graphics
  - View Composition Algebra
  - Grammar of interaction
- Haw it works?

Supplementary materials (demo, notebook): https://iliatimofeev.github.io/PyData2019Altair/

One more visualization library?

Jake VanderPlas The Python Visualization Landscape **PyCon 2017**



Supplementary materials (demo, notebook): https://iliatimofeev.github.io/PyData2019Altair/

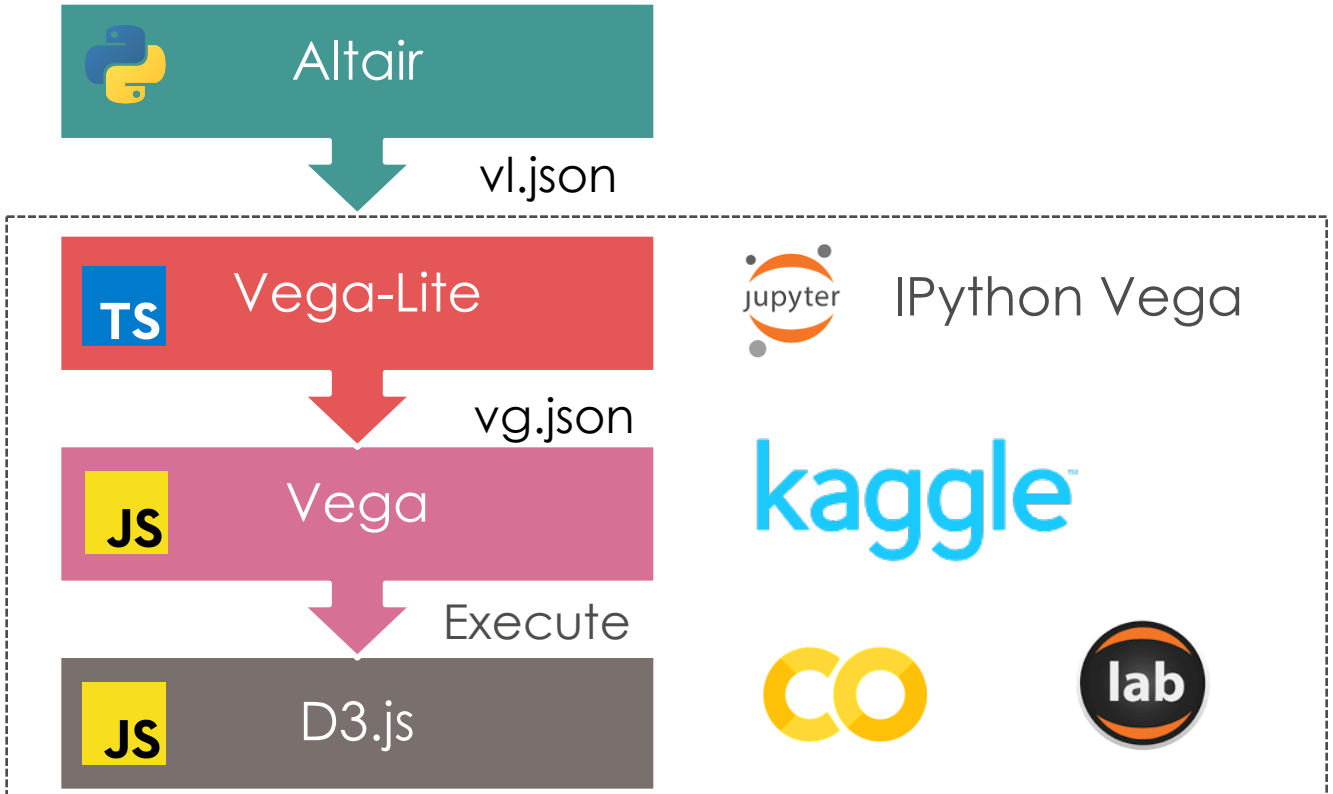
Stack of Altair

Declarative statistical visualization library for Python

High-level grammar for interactive graphics

Visualization grammar
higher-level visualization specification language

JavaScript library for manipulating Data-Driven Documents



Supplementary materials (demo, notebook): https://iliatimofeev.github.io/PyData2019Altair/

Approaches to visualization tool design

Imperative: How?

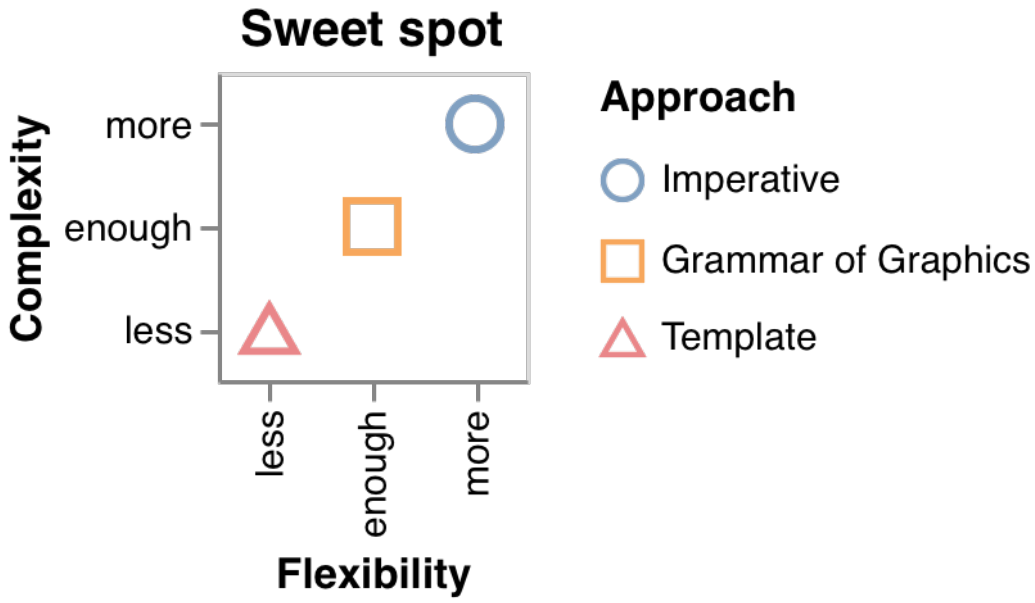
`matplotlib`

Template: Which Template ?

`Seaborn, Plotly`

Grammar of Graphics: What?

`Altair(Vega-Lite), ggplot2(R)`



Supplementary materials (demo, notebook): https://iliatimofeev.github.io/PyData2019Altair/

An example

Hans Rosling several myths about world development TED 2006

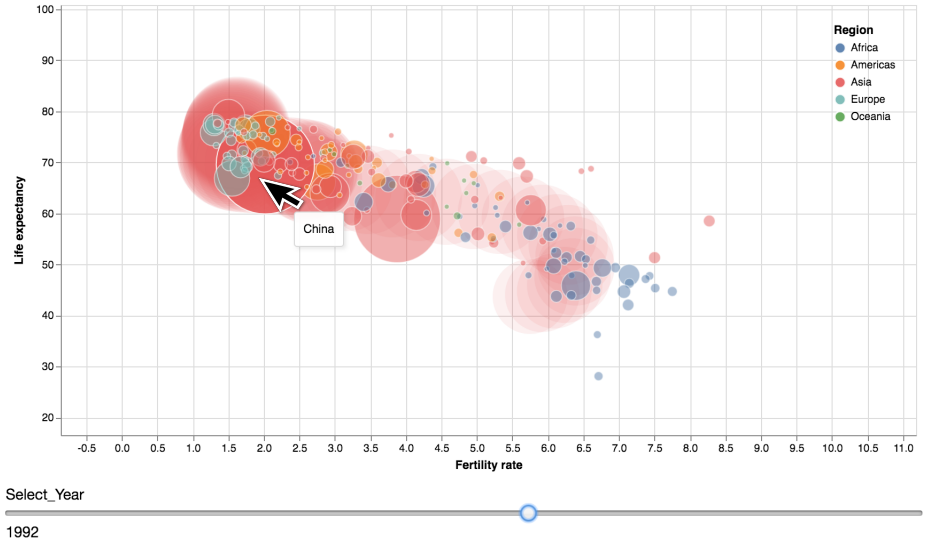
Supplementary materials (demo, notebook):
https://iliatimofeev.github.io/PyData2019Aleair/

40 lines of Python code with Altair and World Bank data

```
base_bubles = (
    alt.Chart()
    .mark_circle(stroke='white', strokeWidth=1)
    .encode(
        alt.X('Fertility rate', scale=alt.Scale(domain=[0, 10])),
        alt.Y('Life expectancy', scale=alt.Scale(domain=[20, 90])),
        alt.Size('Population', legend=None, scale=alt.Scale(range=[20, 10000])),
        alt.Fill('Region', legend=alt.Legend(orient='top-right')),
        alt.Tooltip('Country'),
    )
)

year_input = alt.binding_range(min=1960, max=2016, step=1)
year_sel = alt.selection_single(
    bind=year_input, fields=['Year'], name="Select", empty='none'
)
country_sel = alt.selection_multi(on='mouseover', fields=['Country'], empty='none')
country_sel_highlight = alt.selection_multi(on='mouseover', fields=['Country'])

bubles_full = alt.layer(
    (
        base_bubles.add_selection(year_sel)
        .mark_circle(strokeWidth=0, opacity=0.1)
        .encode(alt.Fill('Region', legend=None))
        .transform_filter(country_sel)
    ),
    (
        base_bubles.transform_filter(year_selection)
        .add_selection(country_sel)
        .add_selection(country_sel_highlight)
        .interactive()
        .encode(
            opacity=alt.condition(
                country_sel_highlight, alt.value(0.9), alt.value(0.5)
            )
        )
    ),
    data=df_data_vis,
).properties(width=800, height=400)
bubles_full
```



One extra line to save as HTML

```
bubles_full.save("demo.html")
```

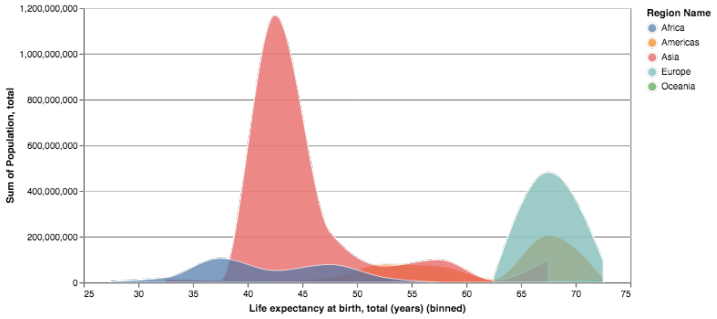
Results:
https://iliatimofeev.github.io/PyData2019Altair/demo.html

Notebook link

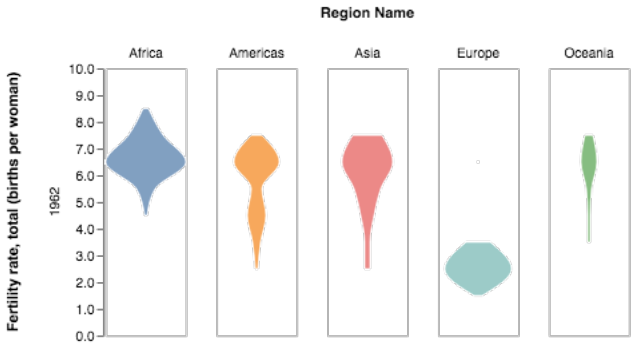
https://iliatimofeev.github.io/PyData2019Altair/

Altair: Python API for Vega-Lite

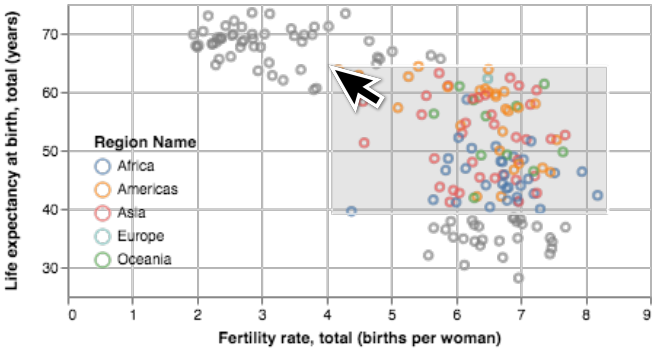
Grammar of Graphics



View Composition Algebra



Grammar of interaction



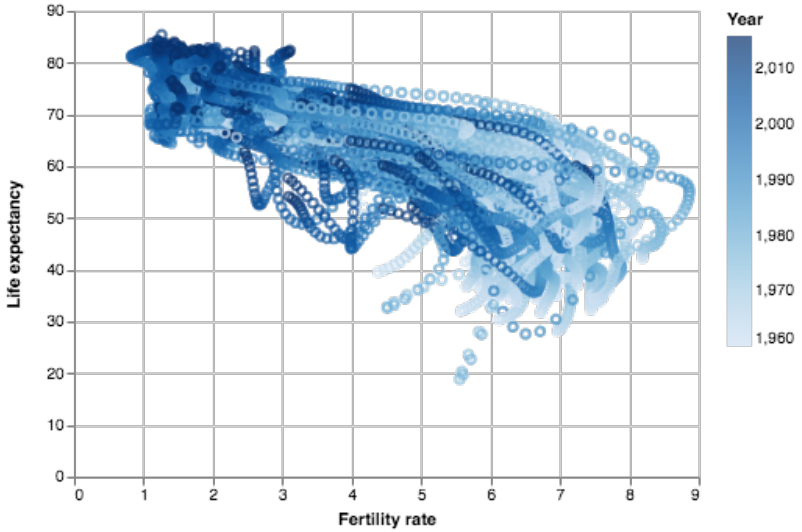
Vega-Lite presented by authors :

Vega Lite: A Grammar of Interactive Graphics - Wongsuphasawat, Moritz, and Satyanarayan Open Viz Conf 2017

Grammar of Graphics

```
(
    alt.Chart( data = df_data_vis)
    .mark_point()
    .encode(
        x = 'Fertility rate',
        y = 'Life expectancy',
        color = 'Year'

    )
)
```

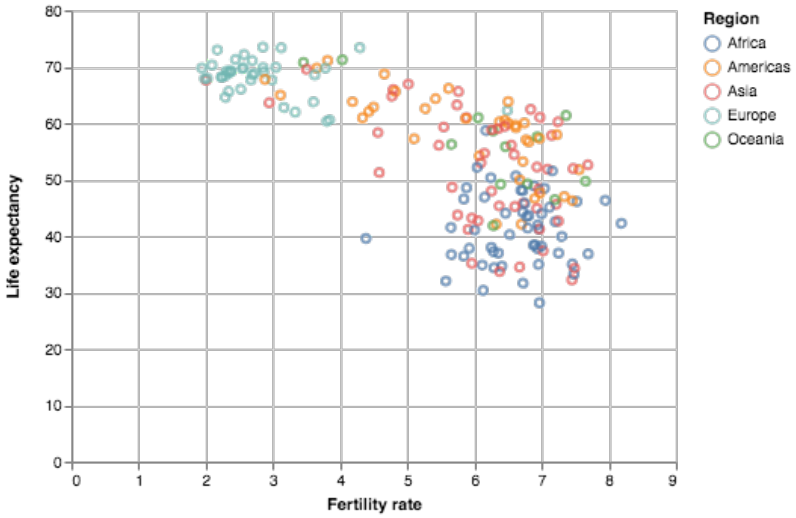


Data → Transformation → Encoding → Mark

Supplementary materials (demo, notebook): https://iliatimofeev.github.io/PyData2019Altair/

Grammar of Graphics Basics

```
1   from altair.expr import datum
2
3   (
4       alt.Chart( data = df_data_vis)
5   +   .transform_filter(datum.Year == 1960)
6       .mark_point()
7       .encode(
8           x = 'Fertility rate',
9           y = 'Life expectancy',
10  ~       color = 'Region'
11
12      )
    )
```



Data → Transformation → Encoding → Mark

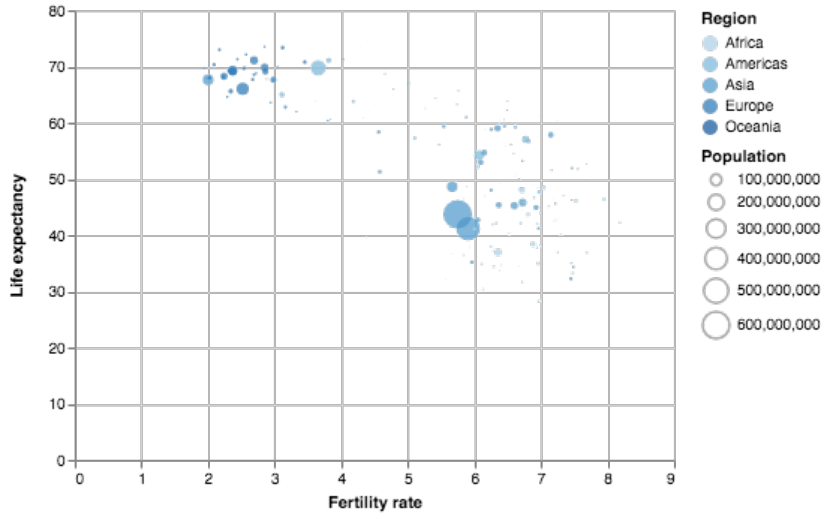
Supplementary materials (demo, notebook): https://iliatimofeev.github.io/PyData2019Altair/

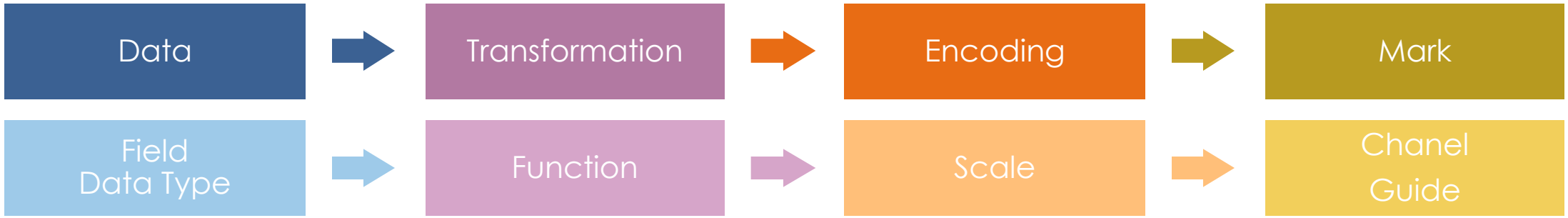
Grammar of Graphics Encoding

```
(
    alt.Chart( data = df_data_vis)
    .transform_filter(datum.Year == 1960)
~   .mark_circle()
    .encode(
        x = 'Fertility rate',
        y = 'Life ',
+       size = 'Population',
~       color = 'Region:O',

    )

)
```

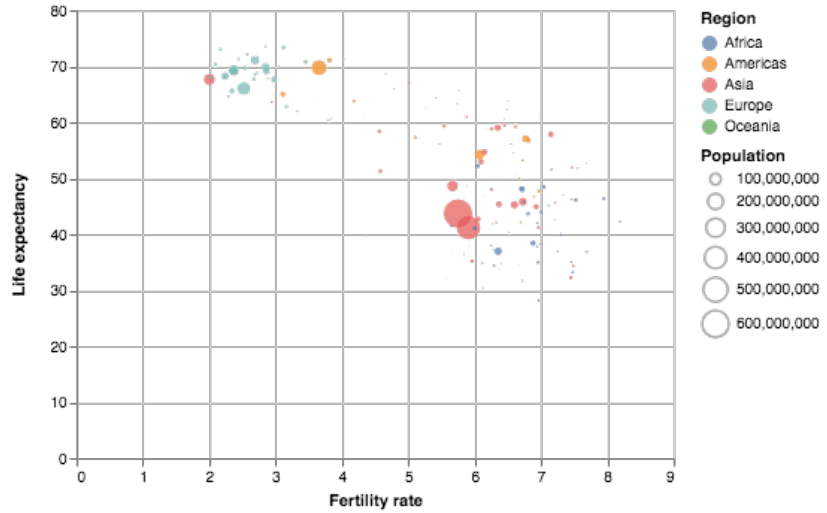




Supplementary materials (demo, notebook): https://iliatimofeev.github.io/PyData2019Altair/

Grammar of Graphics Field Type

```
(
    alt.Chart(data=df_data_vis)
    .transform_filter(datum.Year == 1960)
    .mark_circle()
    .encode(
        x=alt.X('Fertility rate'),
        y=alt.Y(
            shorthand='Life expectancy:Q'
        ),
        size=alt.Size(
            field='Population',
            type='quantitative',
        ),
        color=alt.Color('Region:N'),
    )
)
```

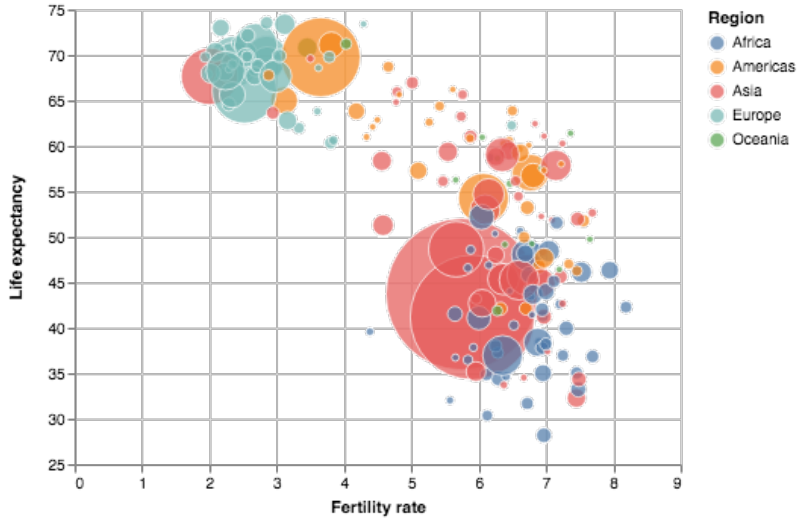


Data Type + Chanel = Default Scale

Field Data Type → Function → Scale → Chanel Guide

Grammar of Graphics Scale

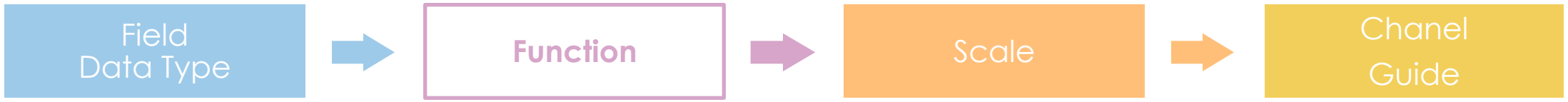
```
1   (
2       alt.Chart( data = df_data_vis)
3       .transform_filter(datum.Year == 1960)
4   ~   .mark_circle(stroke ='white',strokeWidth =1)
5       .encode(
6   ~       alt.X('Fertility rate'),
7   ~       alt.Y(
8               shorthand ='Life expectancy:Q',
9   +           scale = alt.Scale(zero=False),
10              ),
11  ~           alt.Size(
12                  field = 'Population',
13                  type = 'quantitative',
14  +               legend = None,
15  +               scale = alt.Scale(range=[20,10000])
16              ),
17  ~           alt.Fill('Region:N'),
18      )
19  )
```



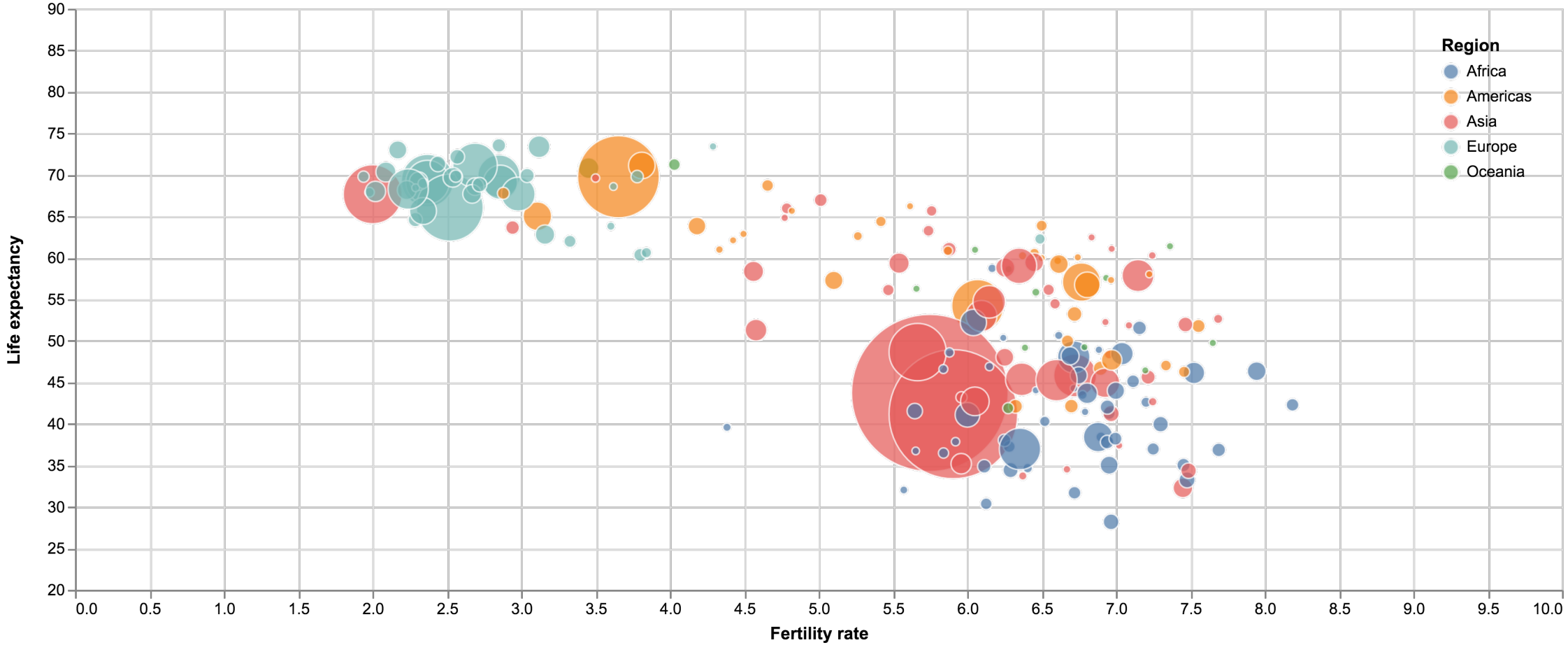
Field Data Type → Function → Scale → Chanel Guide

Grammar of Graphics Final

```
(
alt.Chart(data=df_data_vis, width=950, height=370)
.transform_filter(datum.Year == 1960)
.mark_circle(stroke="white", strokeWidth=1)
.encode(
    alt.X(
        shorthand="Fertility rate",
        scale=alt.Scale(domain=[0, 10])),
    alt.Y(
        shorthand="Life expectancy",
        scale=alt.Scale(domain=[20, 90])),
    alt.Size(
        shorthand="Population:Q",
        legend=None,
        scale=alt.Scale(range=[20, 10000])
    ),
    alt.Fill("Region", legend=alt.Legend(orient="top-right")),
    alt.Tooltip("Country"),
)
)
```



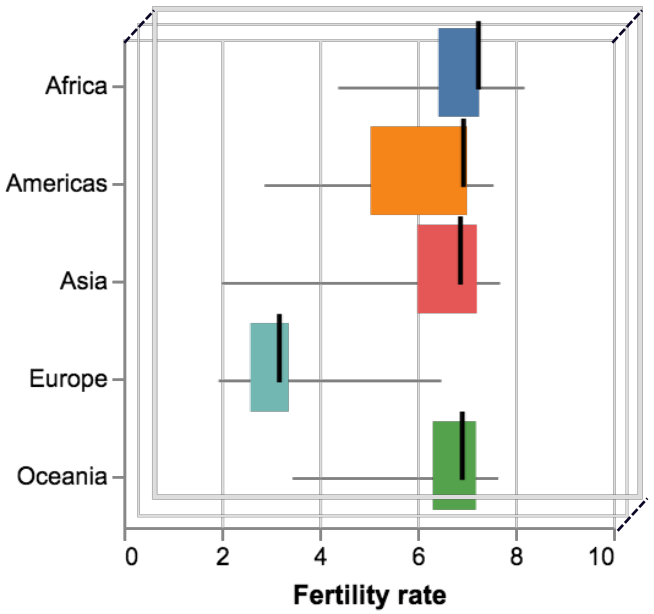
Basic static chart



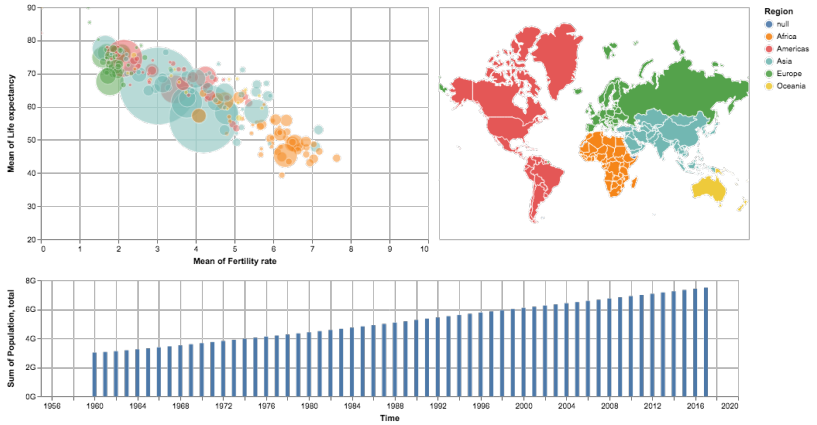
Supplementary materials (demo, notebook): https://iliatimofeev.github.io/PyData2019Altair/

View Composition Algebra

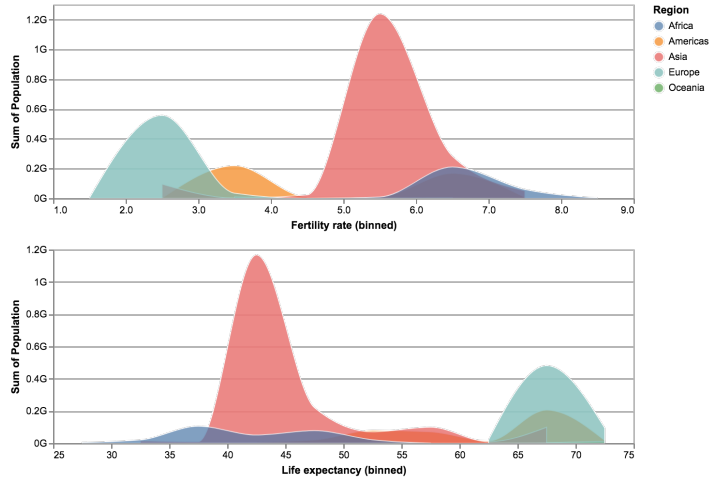
Layer



Concat



Facet/Repeat

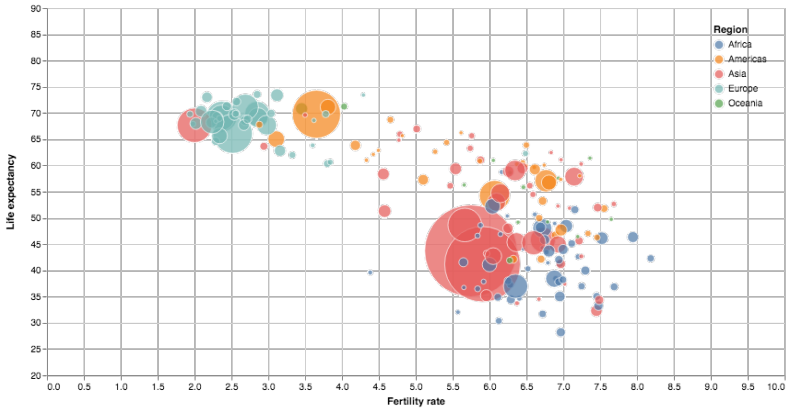


Resolve

Supplementary materials (demo, notebook): https://iliatimofeev.github.io/PyData2019Altair/

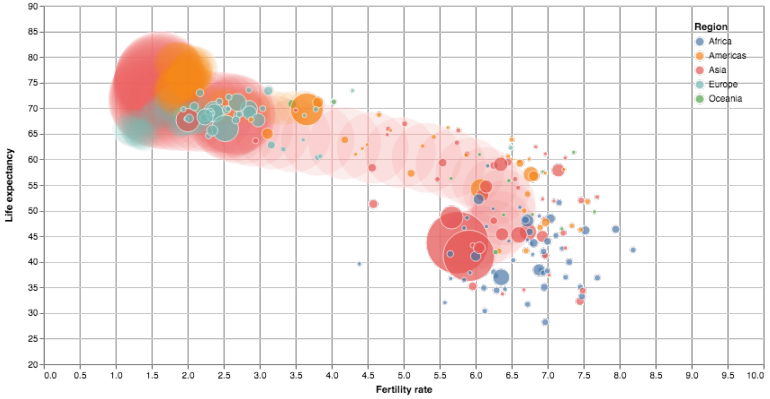
Layer: Prepare

```
~ base_bubles = (
      alt.Chart(data=df_data_vis, width=950, height=370)
–
      .mark_circle(stroke="white", strokeWidth=1)
      .encode(
          alt.X(
              shorthand="Fertility rate",
              scale=alt.Scale(domain=[0, 10]),
          ),
          alt.Y(
              shorthand="Life expectancy",
              scale=alt.Scale(domain=[20, 90]),
          ),
          alt.Size(
              shorthand="Population:Q",
              legend=None,
              scale=alt.Scale(range=[20, 10000]),
          ),
          alt.Fill("Region", legend=alt.Legend(orient="top-right")),
          alt.Tooltip("Country"),
      )
  )
+ base_bubles.transform_filter(datum.Year == 1960)
```



Layer

```
1  + bubles_history = (
2  +     base_bubles.mark_circle(strokeWidth=0, opacity=0.1)
3  +     .encode(alt.Fill("Region Name", legend=None))
4  +     .transform_filter(
5  +         alt.FieldOneOfPredicate(
6  +             field="Country name",
7  +             oneOf=["Russia", "US", "China"],
8  +         )
9  +     )
10 + )
11
12 ~ bubles = base_bubles.transform_filter(datum.Year == 1960)
13
14 + alt.layer(bubles_history, bubles)
```

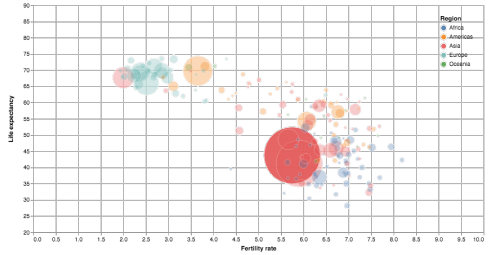


Supplementary materials (demo, notebook): https://iliatimofeev.github.io/PyData2019Altair/

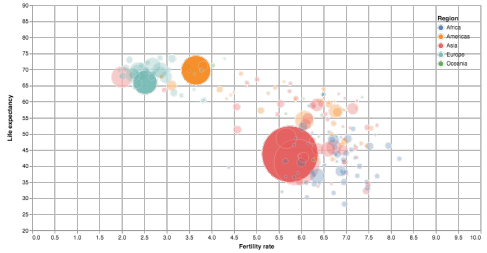
Grammar of interaction

Selection Types

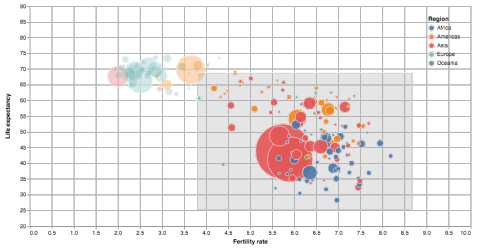
Single

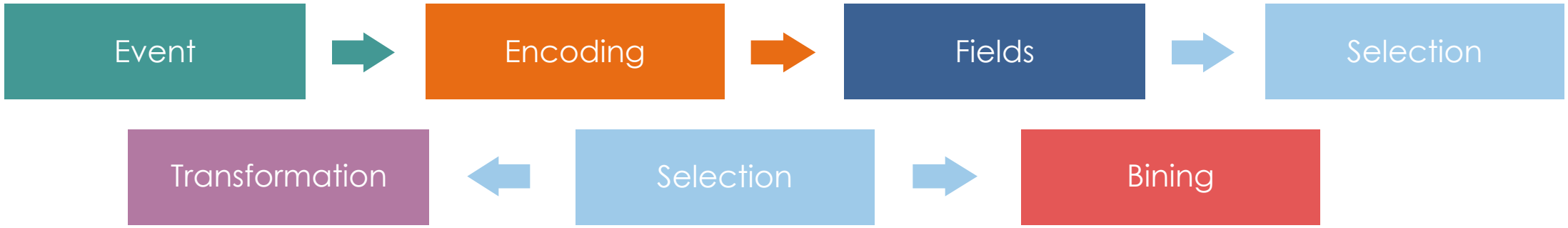


Multi



Interval



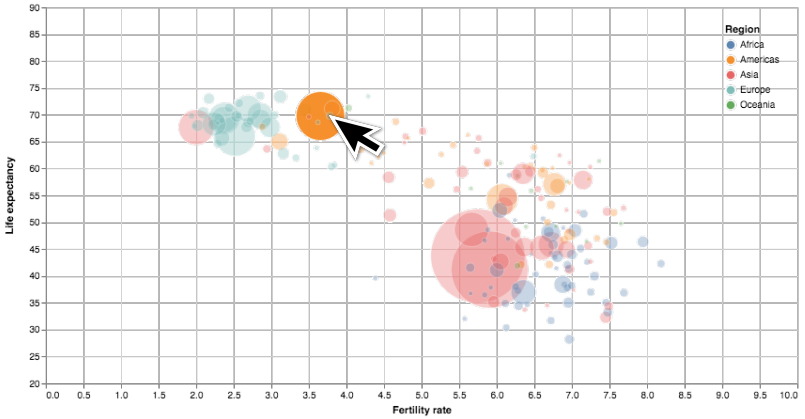


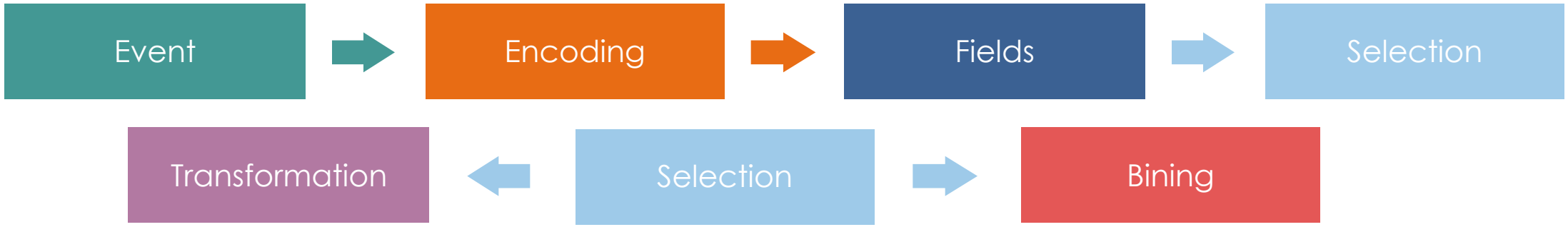
Supplementary materials (demo, notebook): https://iliatimofeev.github.io/PyData2019Altair/

Selection

```
country_sel = alt.selection_single(on='mouseover')

(
    base_bubles.transform_filter(datum.Year == 1960)
    .add_selection(country_sel)
    .interactive()
    .encode(
        opacity=alt.condition(
            country_sel, alt.value(0.9), alt.value(0.3)
        )
    )
)
```

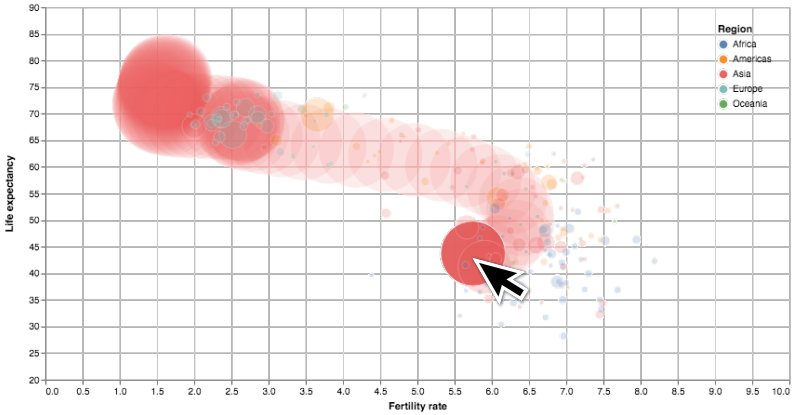




Supplementary materials (demo, notebook): https://iliatimofeev.github.io/PyData2019Altair/

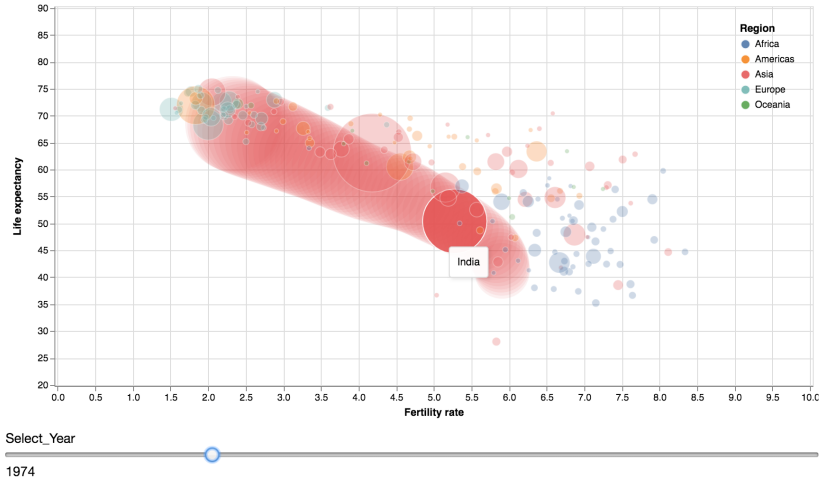
Layers with selection

```
    country_sel = alt.selection_single(on='mouseover')
+   country_sel_hist = alt.selection_multi(
+       on='mouseover', empty='none', fields=['Country']
+   )
+   alt.layer(
+       (
+           base_bubles.mark_circle(strokeWidth=0, opacity=0.1)
+           .encode(alt.Fill('Region', legend=None))
+           .transform_filter(country_sel_hist)
+       ),
        (
            base_bubles.transform_filter(datum.Year == 1960)
+           .add_selection(country_sel_hist)
            .add_selection(country_sel)
            .interactive()
            .encode(
                opacity=alt.condition(
                    country_sel, alt.value(0.9), alt.value(0.5)
                )
            )
        ),
+   )
```

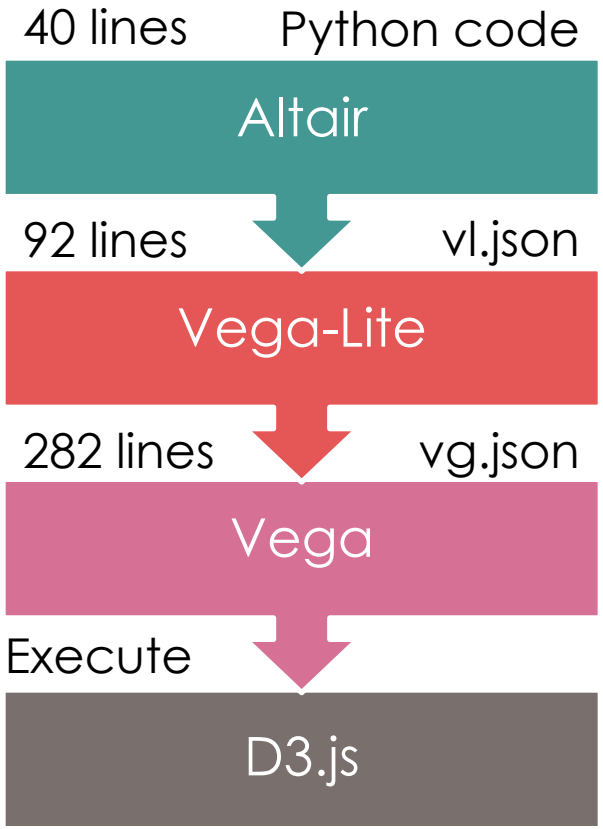


Binding selection

```
 5 +  year_input = alt.binding_range(min=1960, max=2016, step=1)
 6 +  year_selection = alt.selection_single(
 7 +      bind=year_input, fields=['Year'], name="Select", empty='none'
 8 +  )
 9    alt.layer(
10        (
11            base_bubles.mark_circle(strokeWidth=0, opacity=0.
12            .encode(alt.Fill('Region', legend=None))
13            .transform_filter(country_sel_hist)
14 +          .add_selection(year_selection)
15        ),
16        (
17 ~          base_bubles.transform_filter(year_selection)
18            .add_selection(country_sel_hist)
19            .add_selection(country_sel)
20            .interactive()
21            .encode(
22                opacity=alt.condition(
23                    country_sel, alt.value(0.9), alt.value(0.5)
24                )
25            )
26        ),
    )
```



How it works?

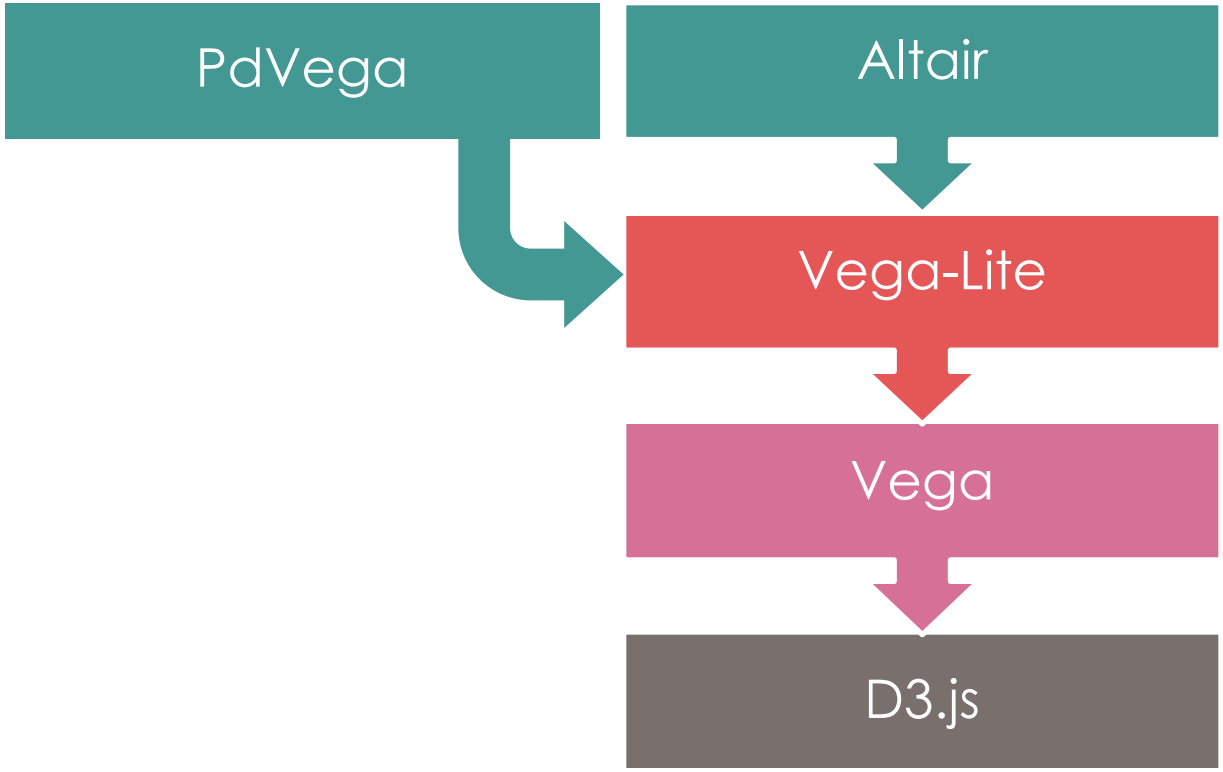


Vega-Lite specification (from first example)

```
{
    "config": {"view": {"width": 400, "height": 300}},
    "data": {
    "url": "word-bank-data.csv",
    "format": {"type": "csv"}
},
    "mark": "point",
    "encoding": {
    "color": {"type": "quantitative", "field": "Year"},
    "x": {"type": "quantitative", "field": "Fertility rate"},
    "y": {"type": "quantitative", "field": "Life expectancy"}
},
"$schema": "https://vega.github.io/schema/vega-lite/v2.6.0.json"
```

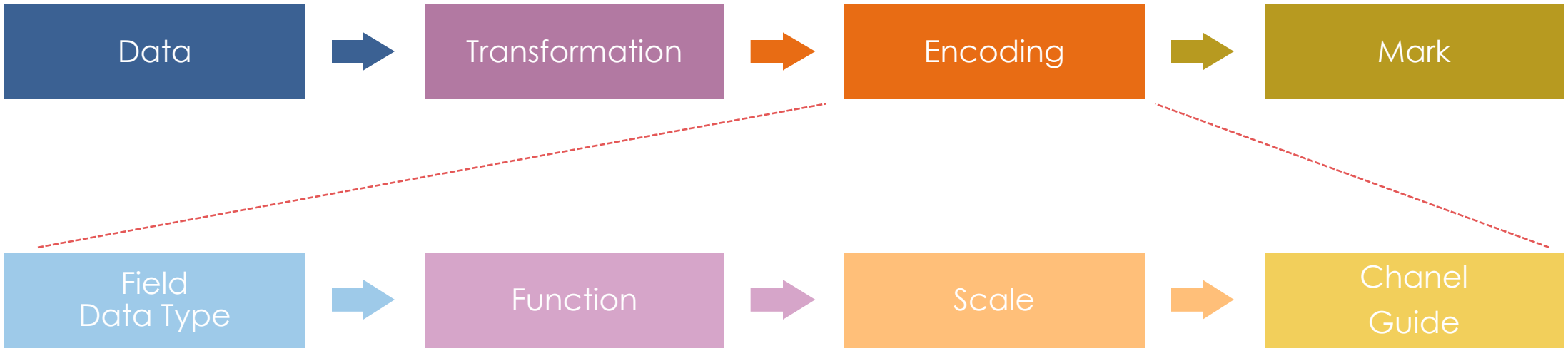
Supplementary materials (demo, notebook): https://iliatimofeev.github.io/PyData2019Altair/

Power of stack



Supplementary materials (demo, notebook): https://iliatimofeev.github.io/PyData2019Altair/

Vega-Lite Grammar of Graphics



Read the article:

Satyanarayan, Arvind & Moritz, Dominik & Wongsuphasawat, Kanit & Heer, Jeffrey. (2016).
Vega-Lite: A Grammar of Interactive Graphics.
IEEE Transactions on Visualization and Computer Graphics. 23. 1-1. 10.1109/TVCG.2016.2599030.

Altair tutorial :

Jake VanderPlas, Exploratory Data Visualization with Altair : https://github.com/altair-viz/altair-tutorial