
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/>

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

Altair: Python API for Vega-Lite Grammar of Graphics

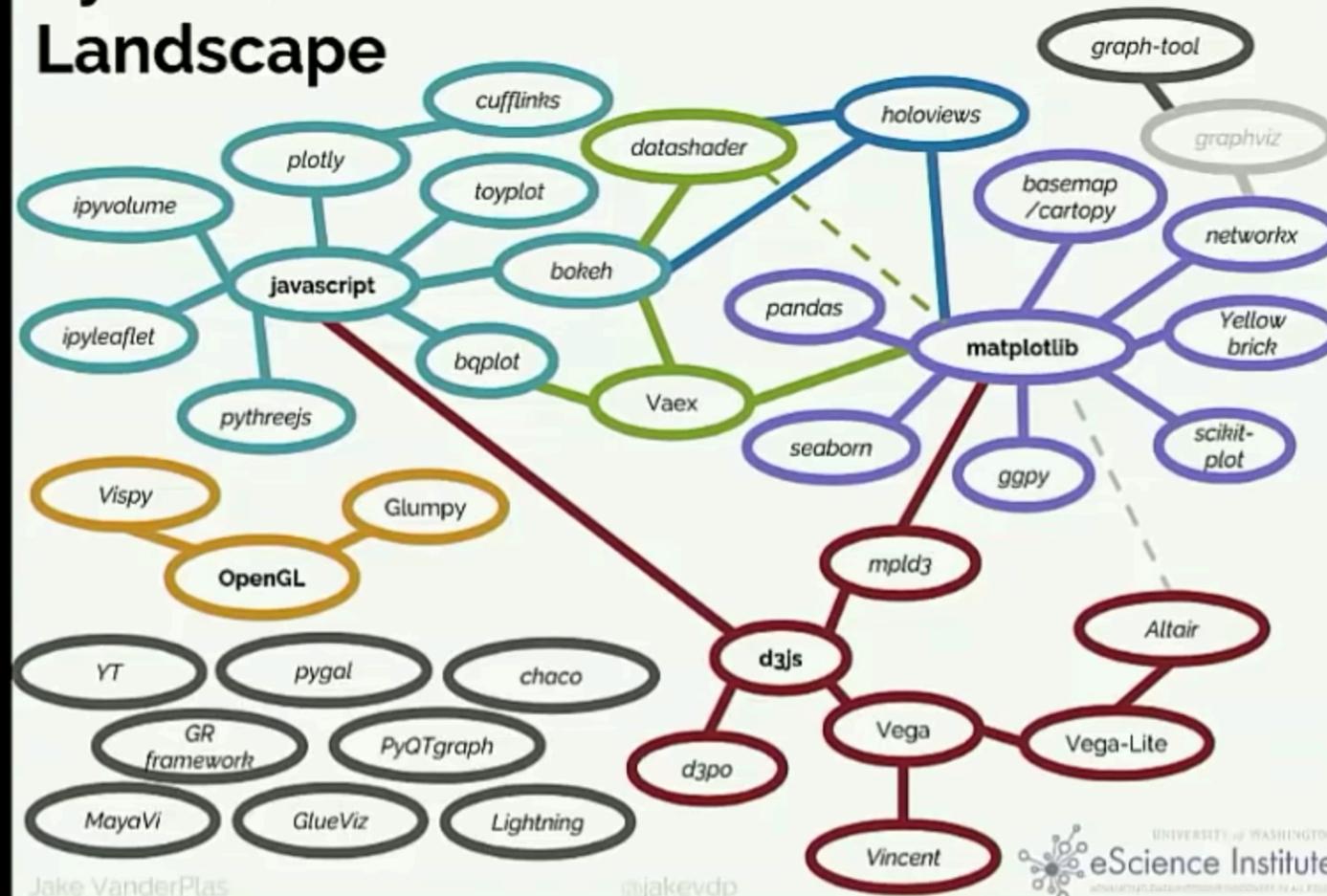
Agenda

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

One more visualization library?

Jake VanderPlas The Python Visualization Landscape [PyCon 2017](#)

Python's Visualization Landscape



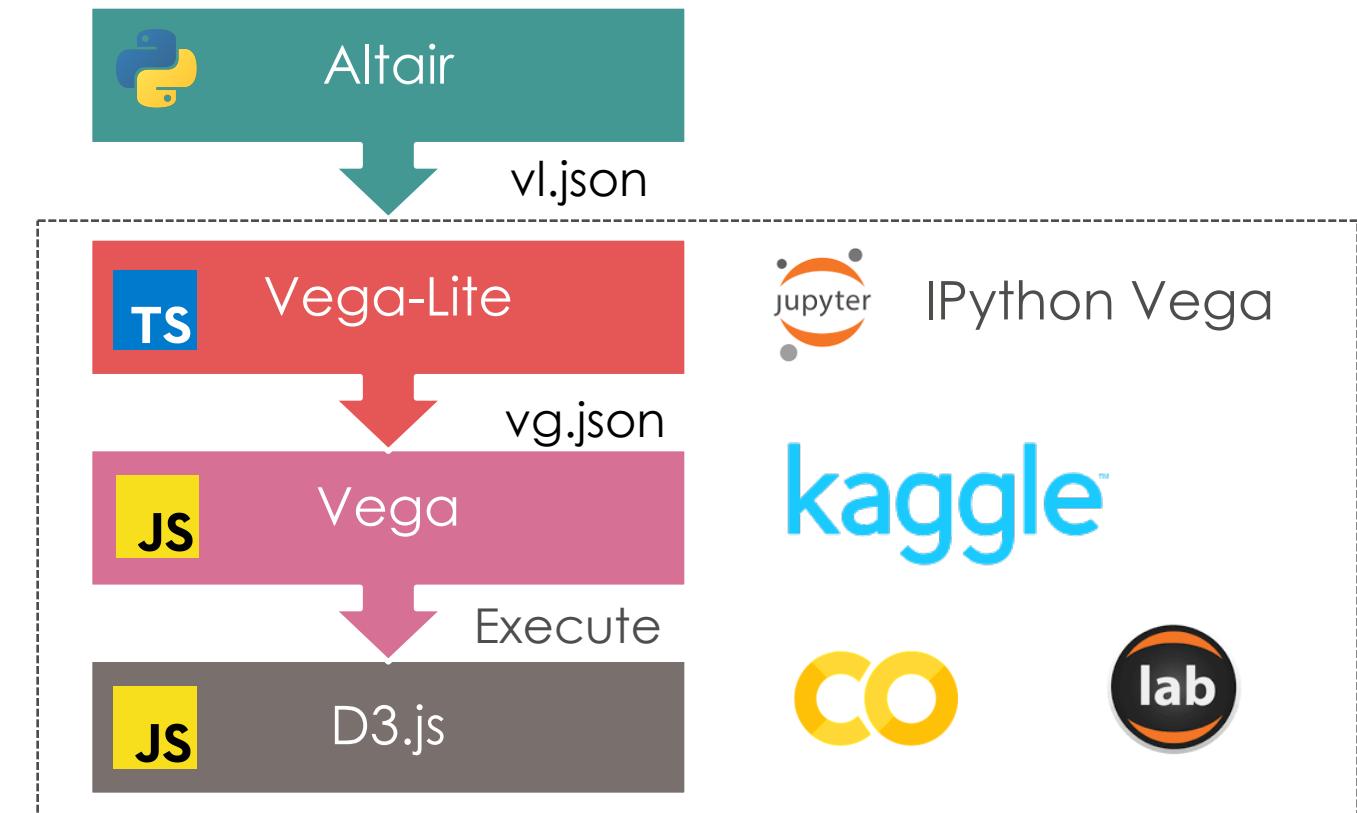
Stack of Altair

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



Approaches to visualization tool design

Imperative: How?

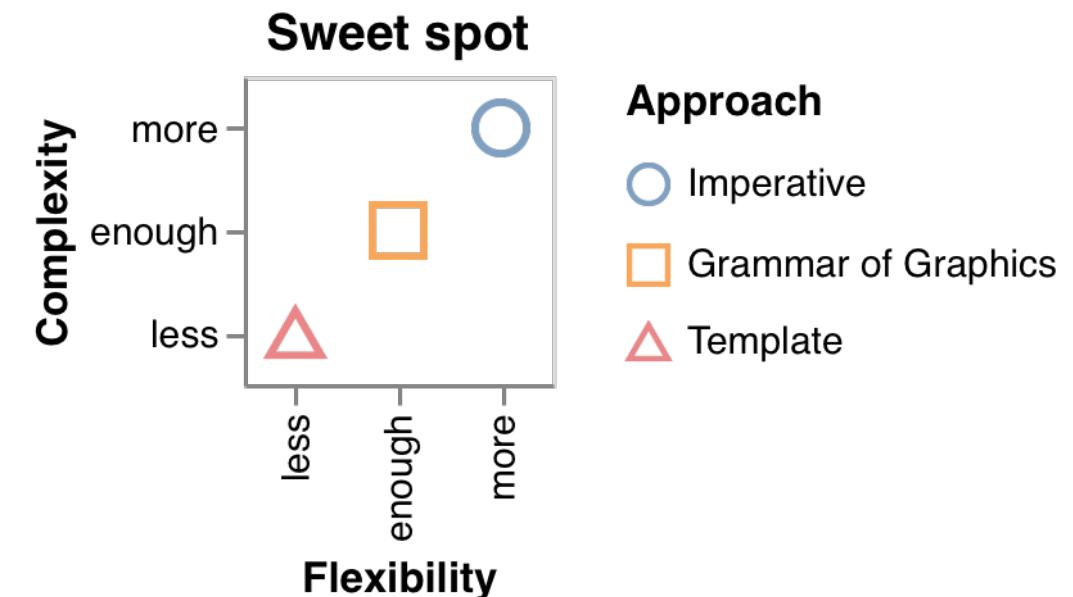
matplotlib

Template: Which Template ?

Seaborn, Plotly

Grammar of Graphics: What?

Altair(Vega-Lite), ggplot2(R)



An example

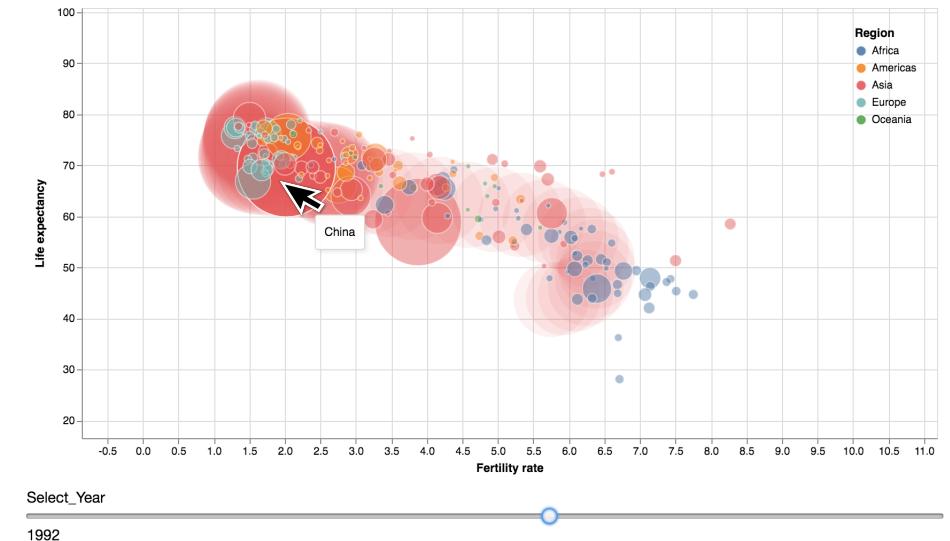
Hans Rosling several myths about world development TED 2006



Supplementary materials (demo, notebook):
<https://iliatimofeev.github.io/PyData2019Atlair/>

40 lines of Python code with Altair and World Bank data

```
1 base_bubbles = (
2     alt.Chart()
3     .mark_circle(stroke='white', strokeWidth=1)
4     .encode(
5         alt.X('Fertility rate', scale=alt.Scale(domain=[0, 10])),
6         alt.Y('Life expectancy', scale=alt.Scale(domain=[20, 90])),
7         alt.Size('Population', legend=None, scale=alt.Scale(range=[20, 10000])),
8         alt.Fill('Region', legend=alt.Legend(orient='top-right')),
9         alt.Tooltip('Country'),
10    )
11 )
12
13 year_input = alt.binding_range(min=1960, max=2016, step=1)
14 year_sel = alt.selection_single(
15     bind=year_input, fields=['Year'], name="Select", empty='none'
16 )
17 country_sel = alt.selection_multi(on='mouseover', fields=['Country'], empty='none')
18 country_sel_highlight = alt.selection_multi(on='mouseover', fields=['Country'])
19
20 bubbles_full = alt.layer(
21     (
22         base_bubbles.add_selection(year_sel)
23         .mark_circle(strokeWidth=0, opacity=0.1)
24         .encode(alt.Fill('Region', legend=None))
25         .transform_filter(country_sel)
26     ),
27     (
28         base_bubbles.transform_filter(year_selection)
29         .add_selection(country_sel)
30         .add_selection(country_sel_highlight)
31         .interactive()
32         .encode(
33             opacity=alt.condition(
34                 country_sel_highlight, alt.value(0.9), alt.value(0.5)
35             )
36         )
37     ),
38     data=df_data_vis,
39 ).properties(width=800, height=400)
bubbles_full
```



One extra line to save as HTML

```
41 bubbles_full.save("demo.html")
```

Results:

<https://iliatimofeev.github.io/PyData2019Altair/demo.html>

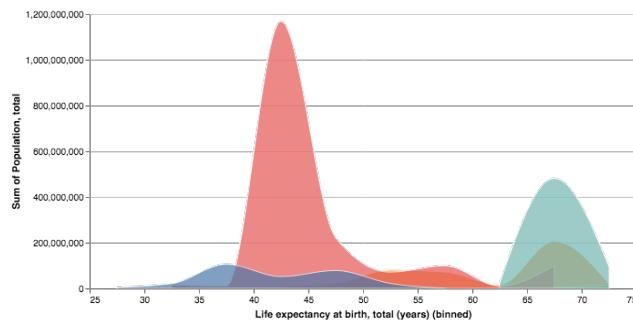
Notebook link

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

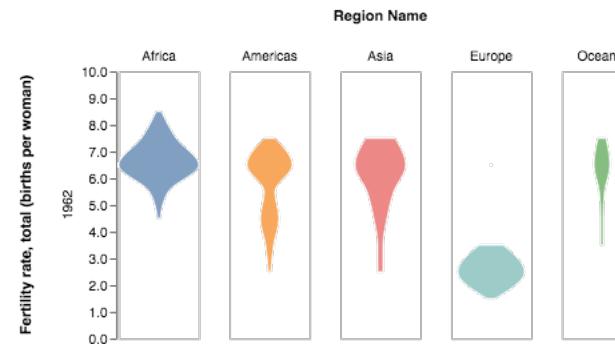


Altair: Python API for Vega-Lite

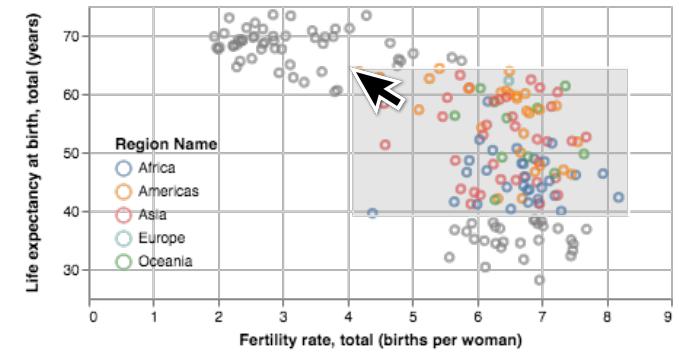
Grammar of Graphics



View Composition Algebra



Grammar of interaction

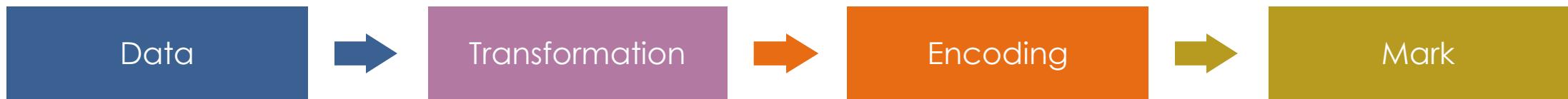
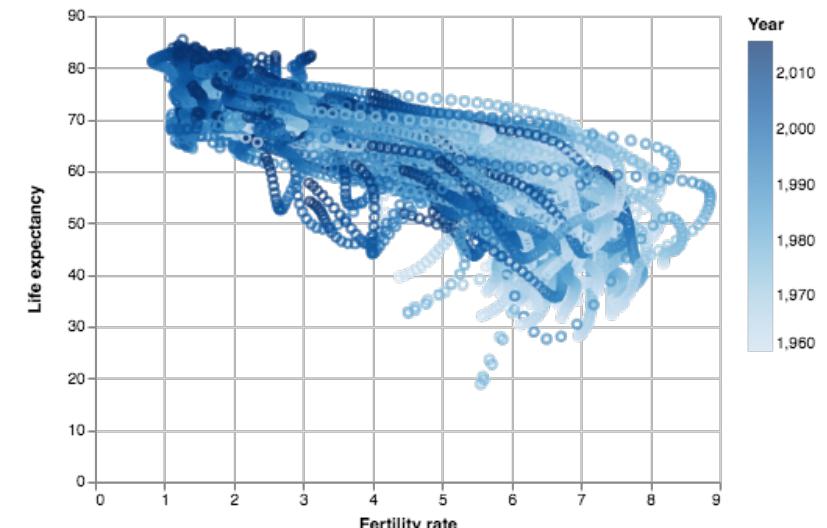


Vega-Lite presented by authors :

Vega Lite: A Grammar of Interactive Graphics - Wongsuphasawat, Moritz, and Satyanarayan [Open Viz Conf 2017](#)

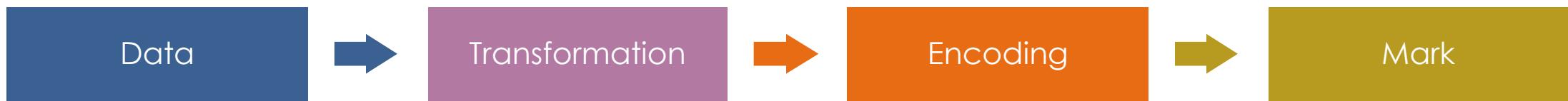
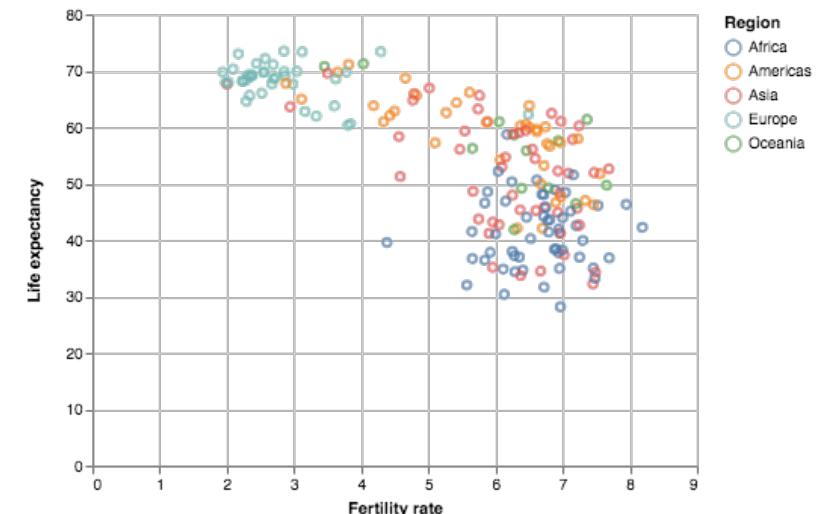
Grammar of Graphics

```
1  (
2      alt.Chart( data = df_data_vis)
3          .mark_point()
4          .encode(
5              x = 'Fertility rate',
6              y = 'Life expectancy',
7              color = 'Year'
8          )
9
10 )
11
12 )
```



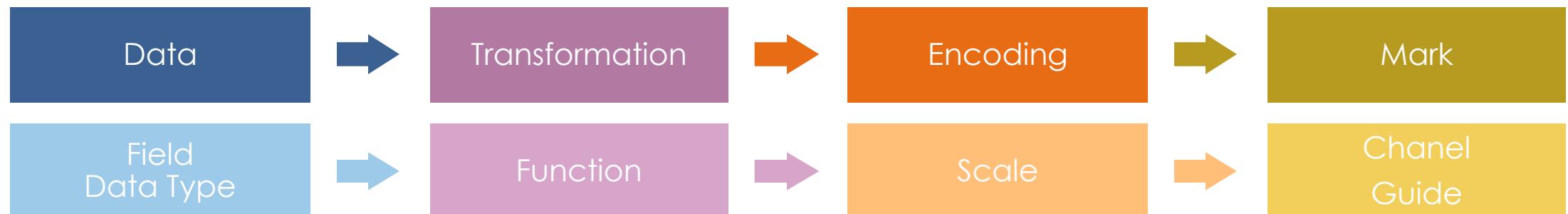
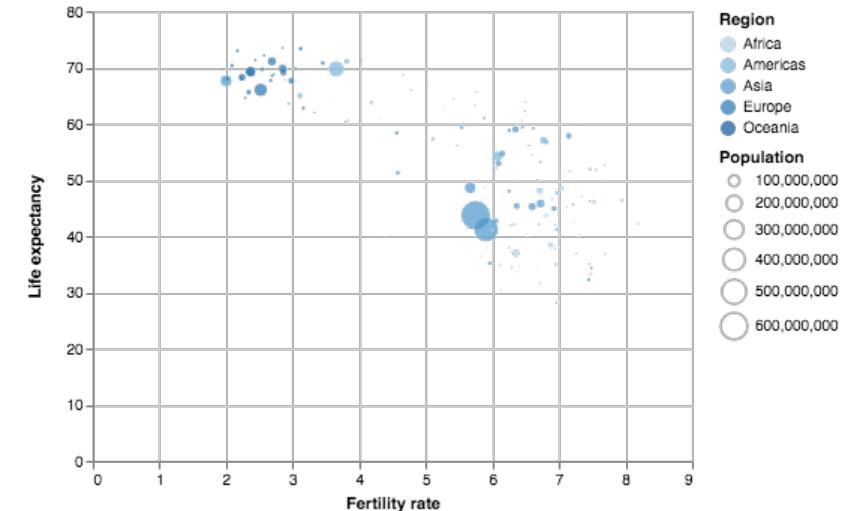
Grammar of Graphics Basics

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



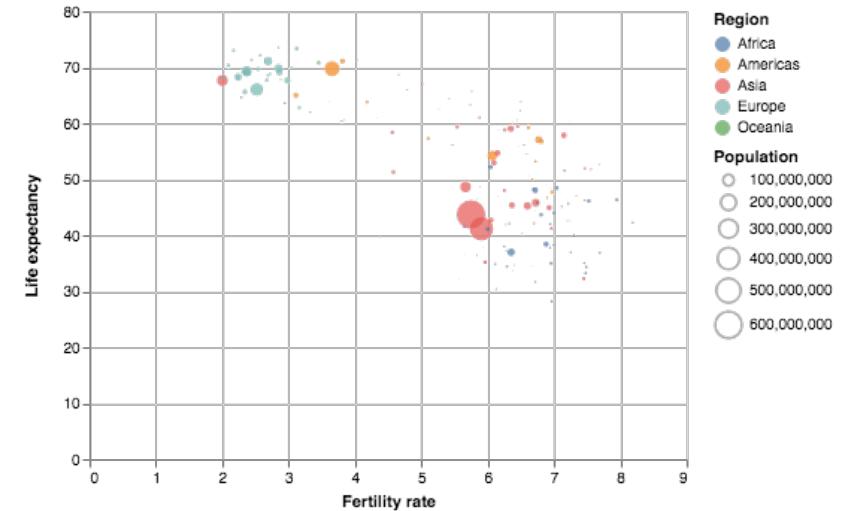
Grammar of Graphics Encoding

```
1  (
2      alt.Chart( data = df_data_vis)
3          .transform_filter(datum.Year == 1960)
4          .mark_circle()
5          .encode(
6              x = 'Fertility rate',
7              y = 'Life ',
8              size = 'Population',
9              color = 'Region:0',
10         )
11     )
12   )
13 )
```



Grammar of Graphics Field Type

```
1  (
2    alt.Chart(data=df_data_vis)
3      .transform_filter(datum.Year == 1960)
4      .mark_circle()
5      .encode(
6        x=alt.X('Fertility rate'),
7        y=alt.Y(
8          shorthand='Life expectancy:Q'
9        ),
10       size=alt.Size(
11         field='Population',
12         type='quantitative',
13       ),
14       color=alt.Color('Region:N'),
15     )
16   )
```



Data Type

Chanel

Default Scale

Field
Data Type

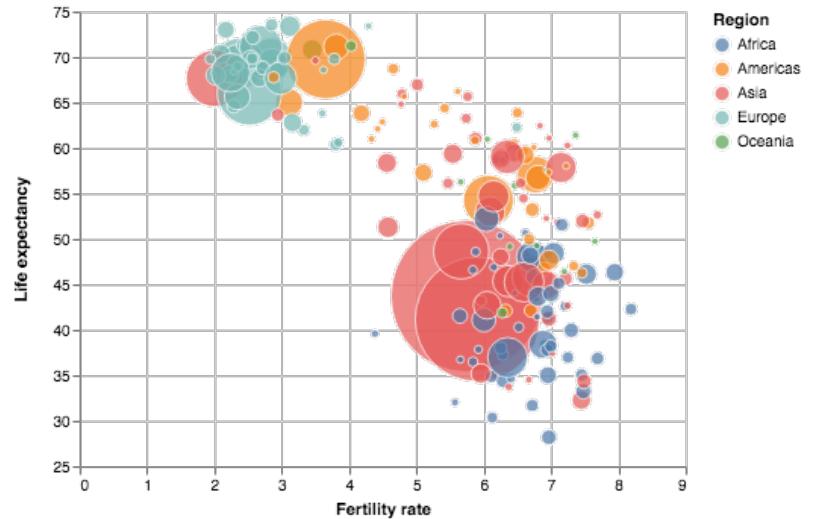
Function

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



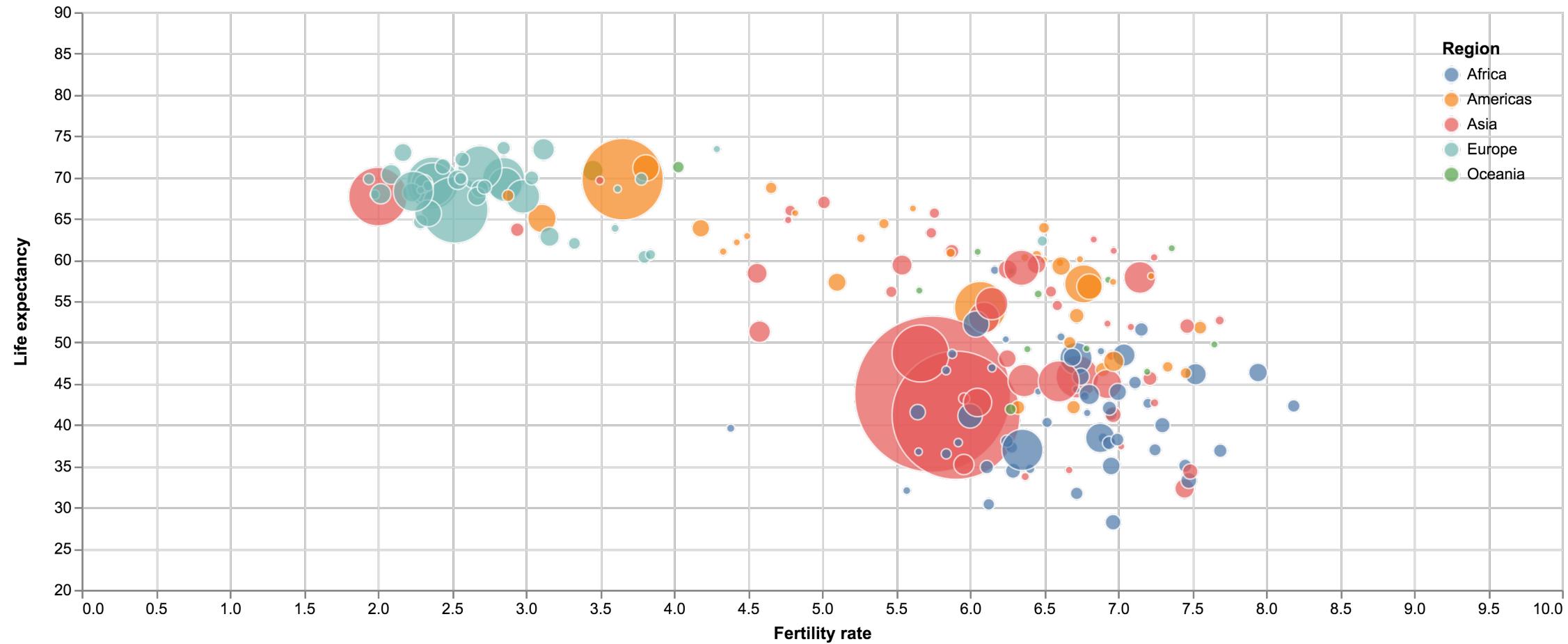
Chanel
Guide

Grammar of Graphics Final

```
1  (
2 ~ alt.Chart(data=df_data_vis, width=950, height=370)
3 .transform_filter(datum.Year == 1960)
4 .mark_circle(stroke="white", strokeWidth=1)
5 .encode(
6   alt.X(
7     shorthand="Fertility rate",
8     scale=alt.Scale(domain=[0, 10])),
9   alt.Y(
10    shorthand="Life expectancy",
11    scale=alt.Scale(domain=[20, 90])),
12   alt.Size(
13     shorthand="Population:Q",
14     legend=None,
15     scale=alt.Scale(range=[20, 10000]))
16 ),
17 alt.Fill("Region", legend=alt.Legend(orient="top-right")),
18 alt.Tooltip("Country"),
19 )
20 )
```



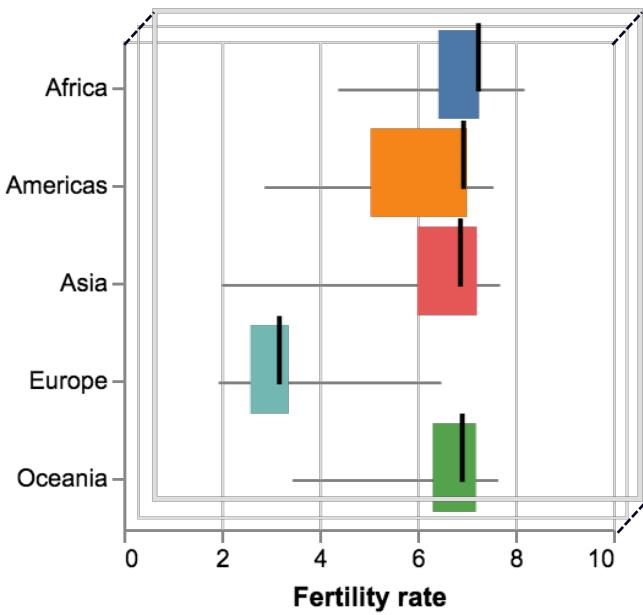
Basic static chart



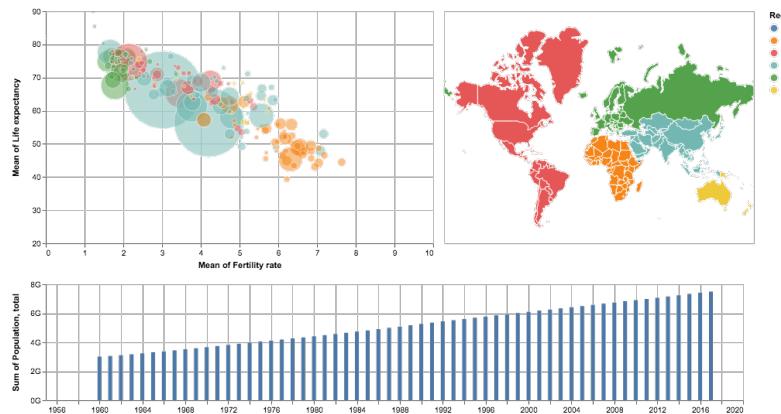
Supplementary materials (demo, notebook): <https://iliatimofeev.github.io/PyData2019Atlair/>

View Composition Algebra

Layer

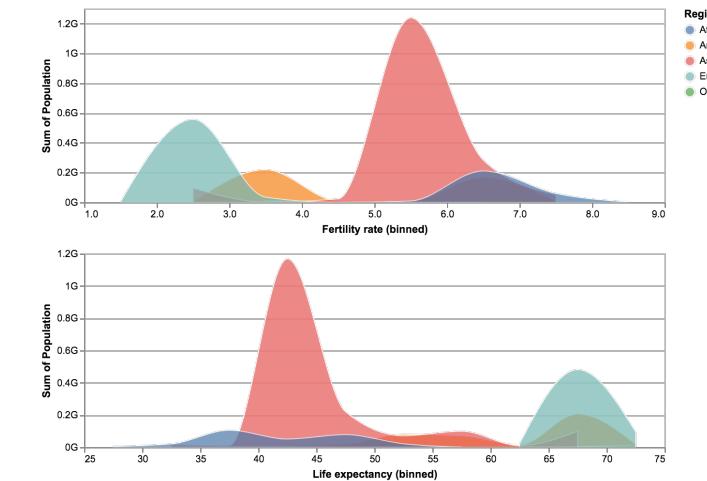


Concat



Resolve

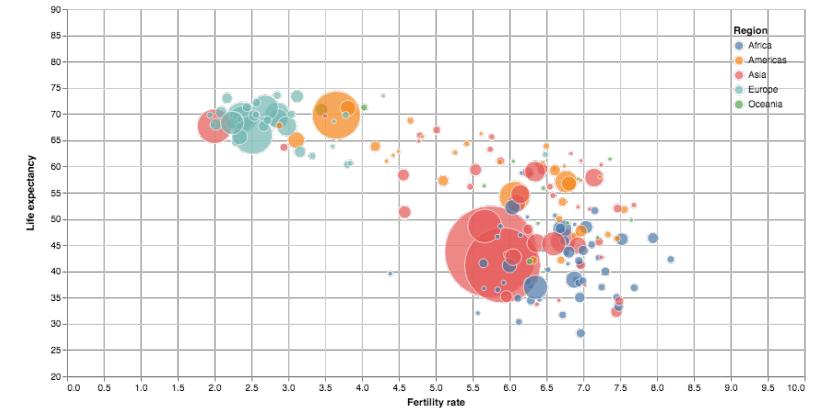
Facet/Repeat



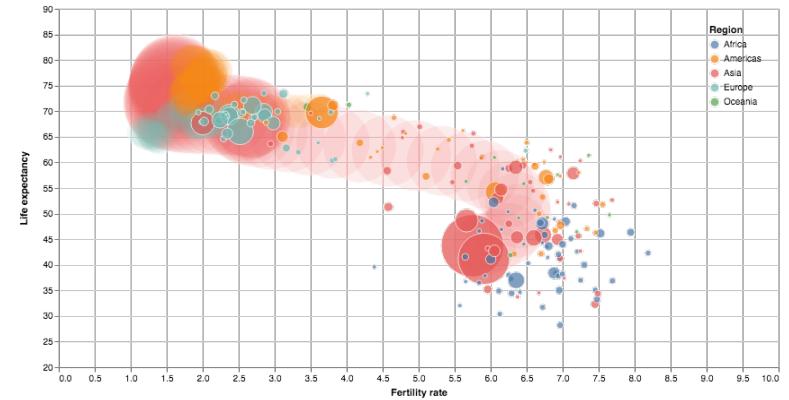
Supplementary materials (demo, notebook): <https://iliatimofeev.github.io/PyData2019Atlair/>

Layer: Prepare

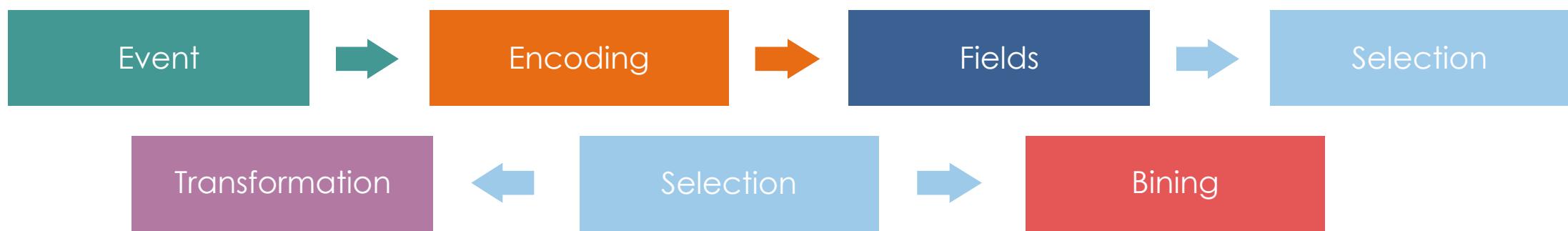
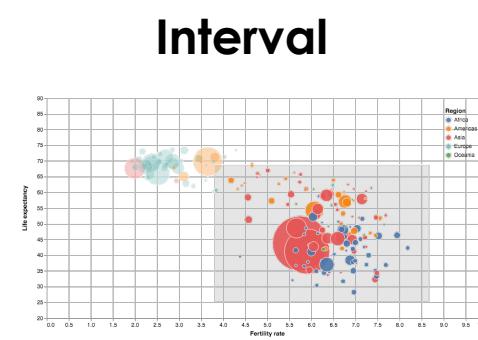
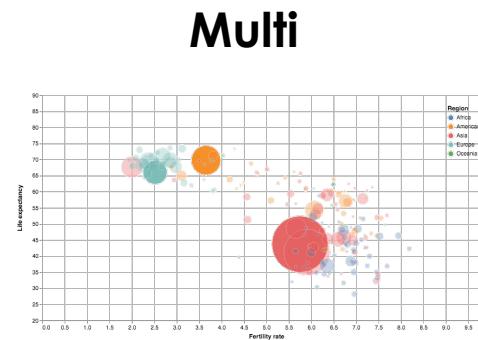
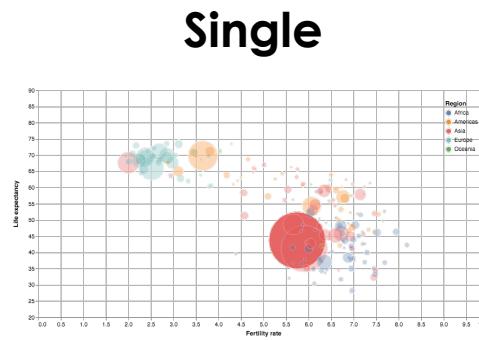
```
1 ~ base_bubbles = (
2     alt.Chart(data=df_data_vis, width=950, height=370)
3     - .mark_circle(stroke="white", strokeWidth=1)
4     .encode(
5         alt.X(
6             shorthand="Fertility rate",
7             scale=alt.Scale(domain=[0, 10]),
8         ),
9         alt.Y(
10            shorthand="Life expectancy",
11            scale=alt.Scale(domain=[20, 90]),
12        ),
13         alt.Size(
14             shorthand="Population:Q",
15             legend=None,
16             scale=alt.Scale(range=[20, 10000]),
17         ),
18         alt.Fill("Region", legend=alt.Legend(orient="top-right")),
19         alt.Tooltip("Country"),
20     )
21 )
22 )
23 + base_bubbles.transform_filter(datum.Year == 1960)
```



```
1 + bubbles_history = (
2 +     base_bubbles.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 ~ bubbles = base_bubbles.transform_filter(datum.Year == 1960)
13
14 + alt.layer(bubbles_history, bubbles)
```

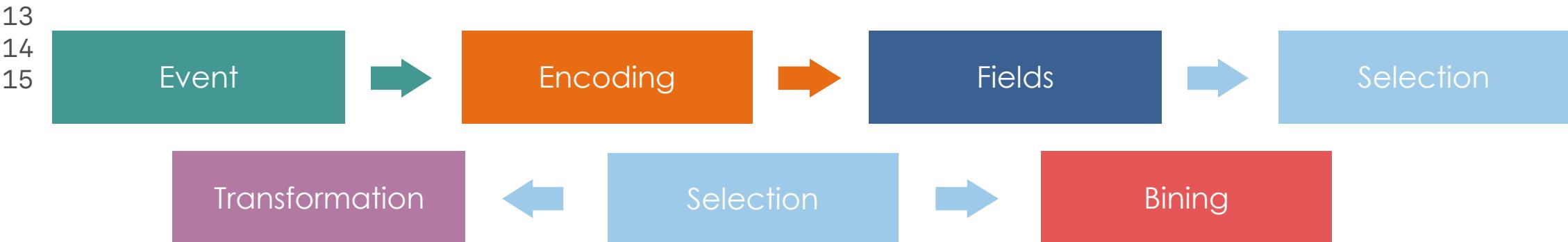
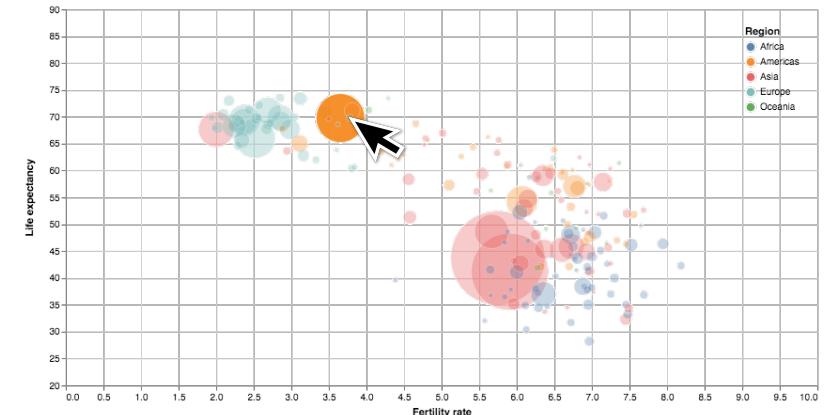


Selection Types



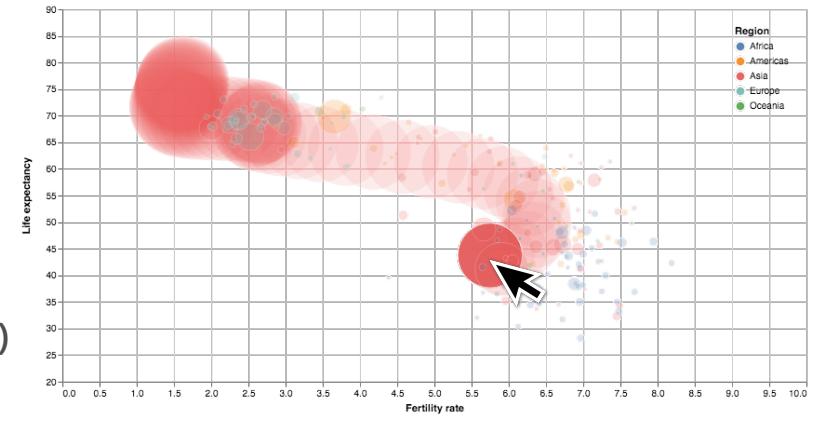
Selection

```
1 + country_sel = alt.selection_single(on='mouseover')
2 +
3 ( base_bubbles.transform_filter(datum.Year == 1960)
4 .add_selection(country_sel)
5 .interactive()
6 .encode(
7     opacity=alt.condition(
8         country_sel, alt.value(0.9), alt.value(0.3)
9     )
10 )
11 )
12 )
```



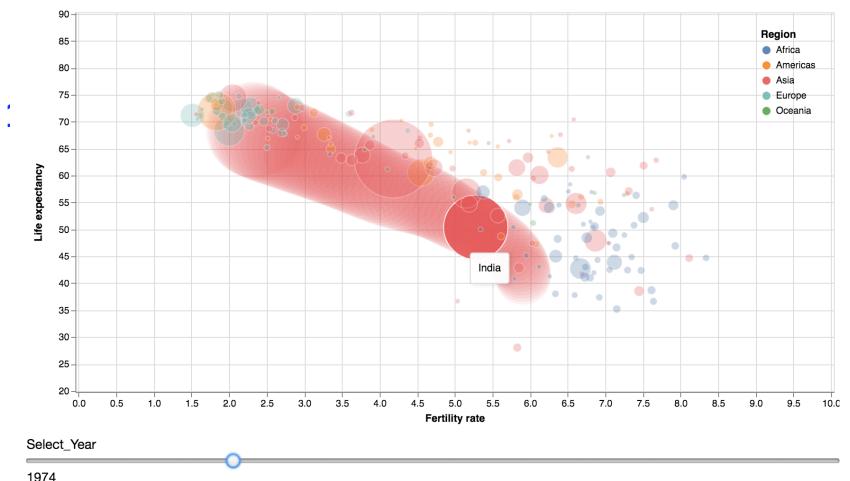
Layers with selection

```
1 country_sel = alt.selection_single(on='mouseover')
2 +
3 +     country_sel_hist = alt.selection_multi(
4 +         on='mouseover', empty='none', fields=['Country']
5 +     )
6 +     alt.layer(
7 +         (
8 +             base_bubbles.mark_circle(strokeWidth=0, opacity=0.1)
9 +             .encode(alt.Fill('Region', legend=None))
10 +            .transform_filter(country_sel_hist)
11 +        ),
12 +
13 +        (
14 +            base_bubbles.transform_filter(datum.Year == 1960)
15 +            .add_selection(country_sel_hist)
16 +            .add_selection(country_sel)
17 +            .interactive()
18 +            .encode(
19 +                opacity=alt.condition(
20 +                    country_sel, alt.value(0.9), alt.value(0.5)
21 +                )
22 +            )
23 +        )
24 +    )
```

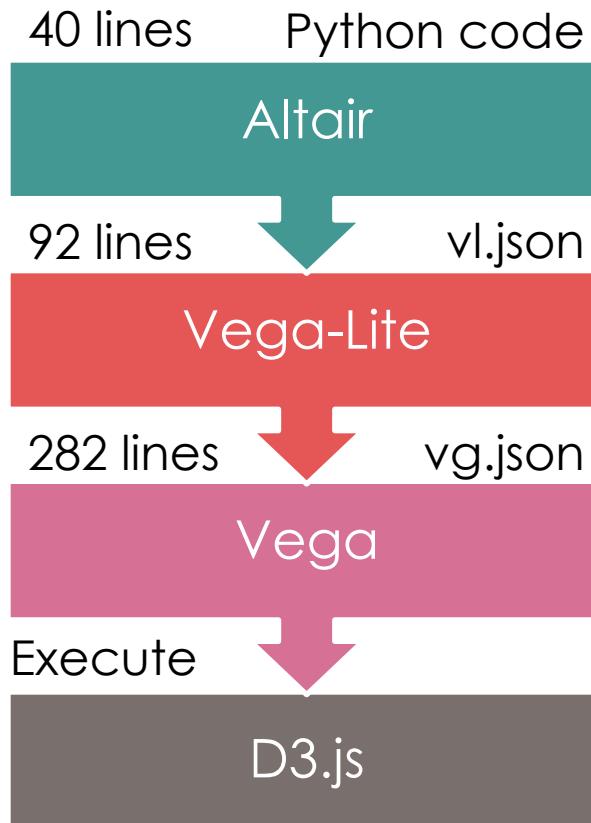


Binding selection

```
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_bubbles.mark_circle(strokeWidth=0, opacity=0.1)
12     .encode(alt.Fill('Region', legend=None))
13     .transform_filter(country_sel_hist)
14     .add_selection(year_selection)
15 ),
16 (
17     ~base_bubbles.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 ),
```



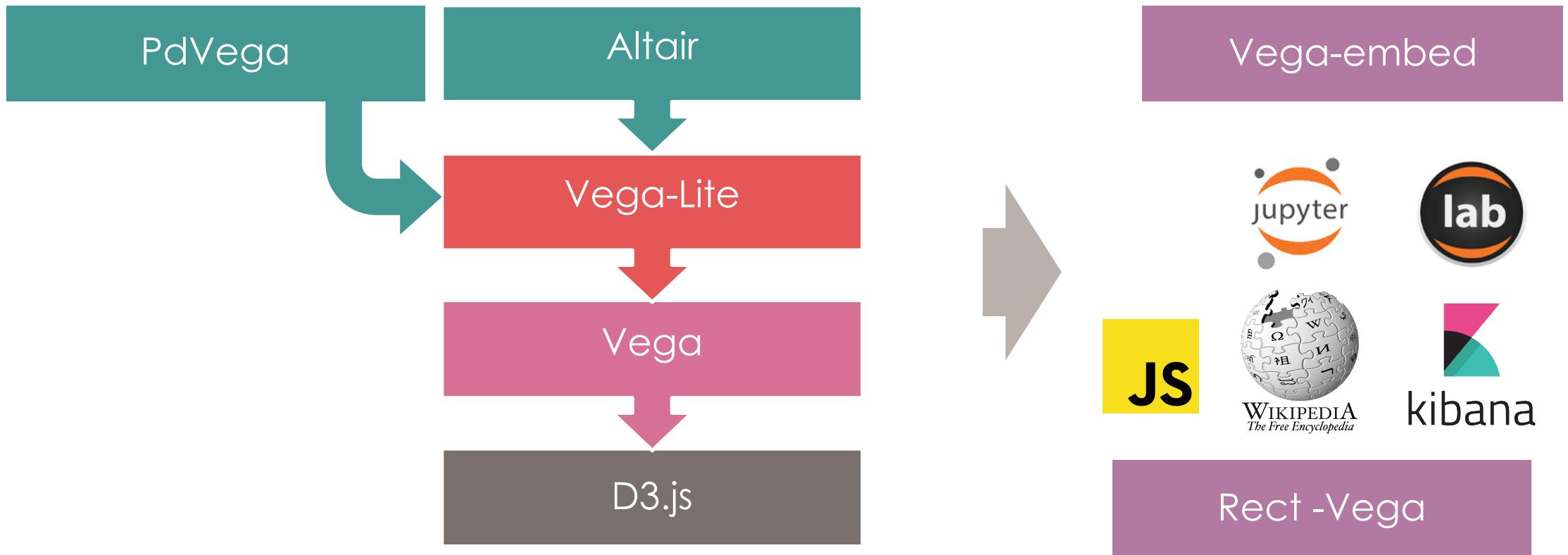
How it works?



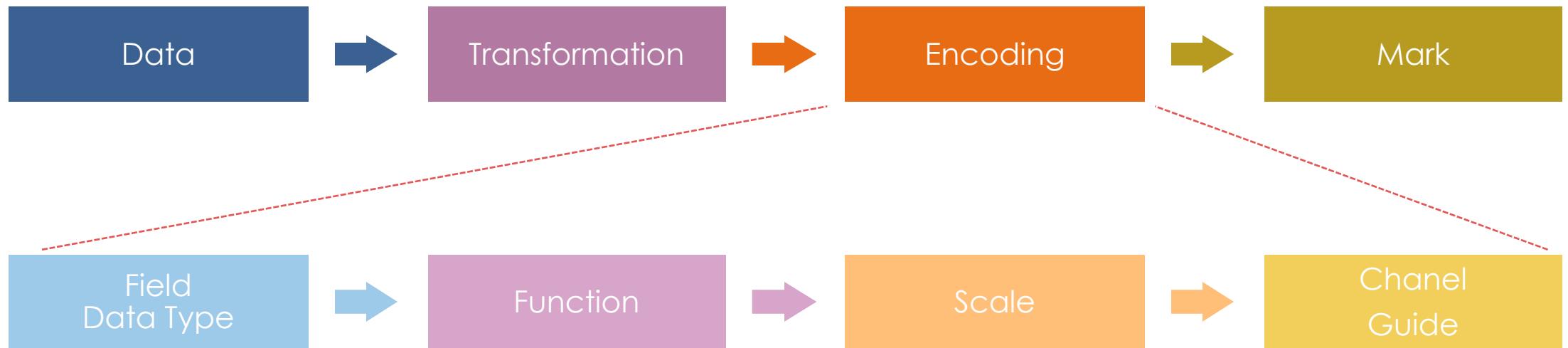
Vega-Lite specification (from first example)

```
{  
  "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"
```

Power of stack



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>