TBD

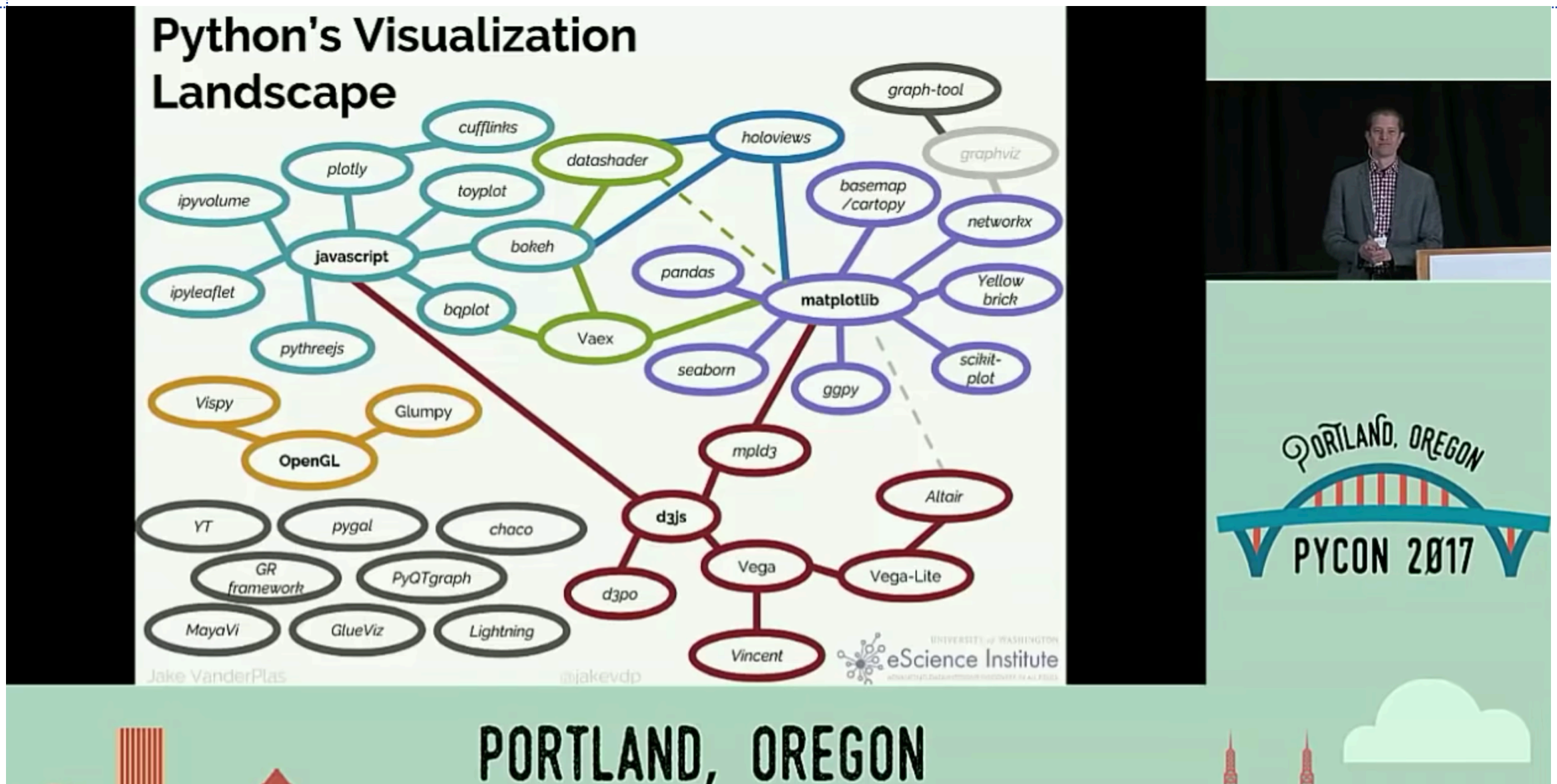
Altair is a declarative statistical visualization library for Python

- API for grammar of graphics (like ggplot2)
- with interactive graphics support
- based on powerful stack of Vega-Lite/Vega/d3.js

Ilia Timofeev, Senior partner, TTS Consulting

One more visualization library?

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



Approaches to visualization tool design

Imperative: How?

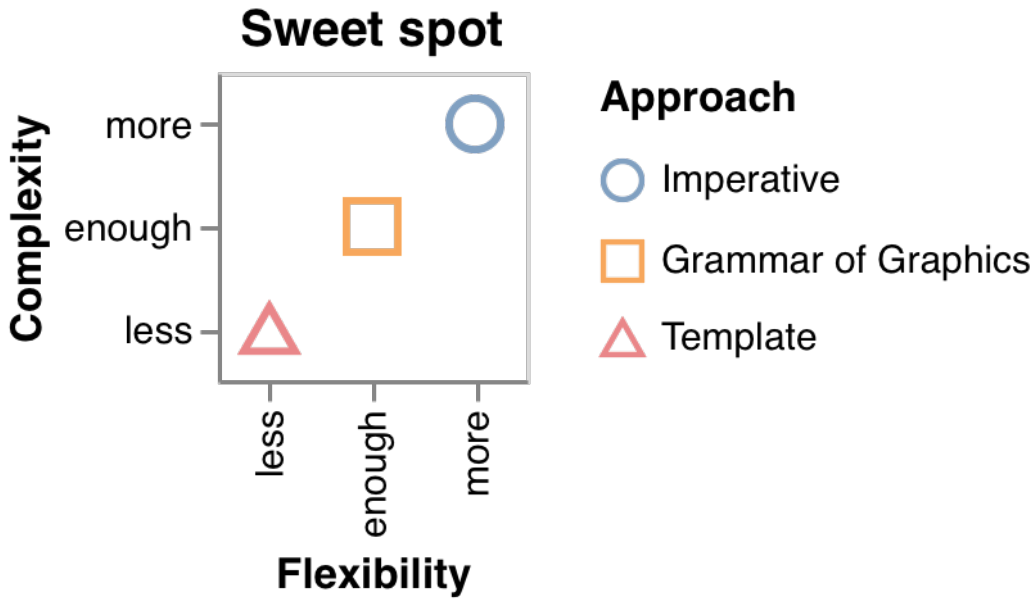
`matplotlib`

Template: Which Template ?

`Seaborn, Plotly`

Grammar of Graphics: What?

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

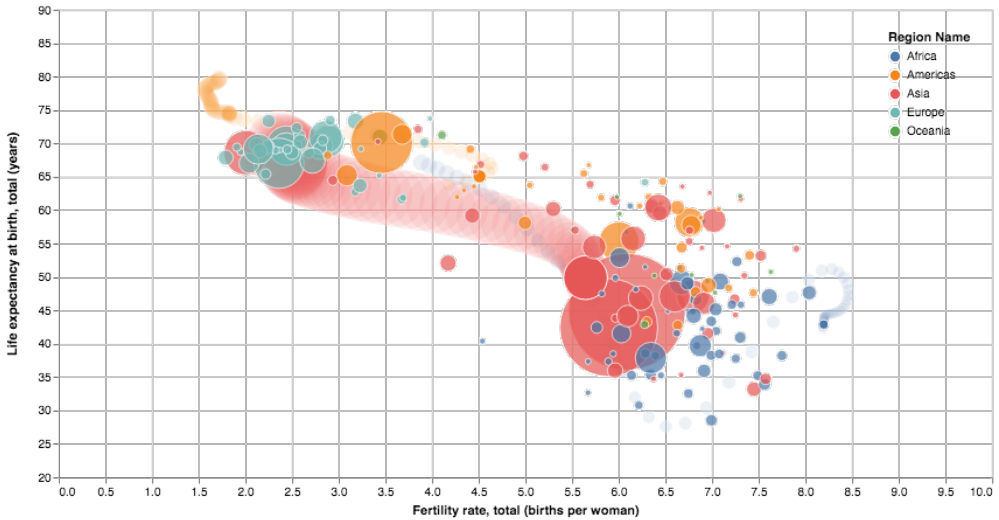


An example

Hans Rosling several myths about world development TED 2006

40 lines of Altair code and World Bank data

```
year_slider = alt.binding_range(min=1960, max=2016, step=1)
year_selection = alt.selection_single(
    bind=year_slider, fields=["Time"], name="Select", empty="none"
)
name_selection = alt.selection_multi(fields=["CLDR display name"], empty="none")
base_bubbles = (
    alt.Chart()
    .mark_circle(stroke="white", strokeWidth=1)
    .encode(
        alt.X(
            "Fertility rate, total (births per woman)",
            scale=alt.Scale(zero=False, domain=[0, 10]),
        ),
        alt.Y(
            "Life expectancy at birth, total (years)",
            scale=alt.Scale(zero=False, domain=[20, 90]),
        ),
        alt.Size("Population, total", legend=None, scale=alt.Scale(range=[20, 10000])),
        alt.Fill("Region Name", legend=None),
        tooltip=["CLDR display name", "Time:N"],
    )
)
bubble_chart = alt.layer(
    (
        base_bubbles.mark_circle(opacity=0.1, strokeWidth=0).transform_filter(
            name_selection
        )
    ),
    (
        base_bubbles.mark_circle(stroke="white", strokeWidth=1)
        .transform_filter(year_selection)
        .)
        .add_selection(name_selection)
        .encode(
            fill="Region Name",
            opacity=alt.condition(name_selection, alt.value(1), alt.value(0.7)),
        )
    ),
    data=df_data_vis
)
bubble_chart.properties(width = 800,height=400).save("demo.html")
```

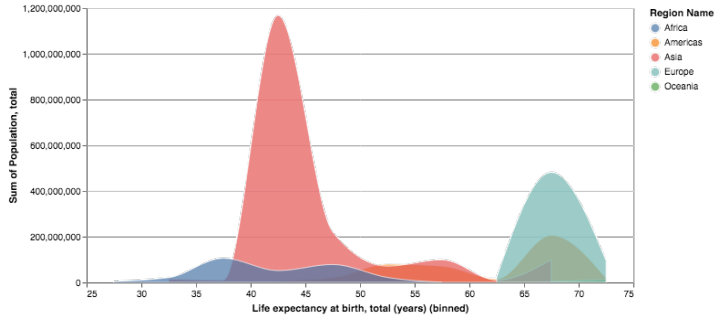


Live demo: demo.html

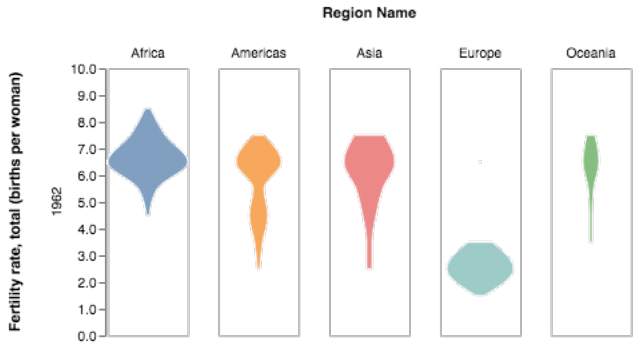
jupyter CO Run code in notebook

Altair: Python API for Vega-Lite

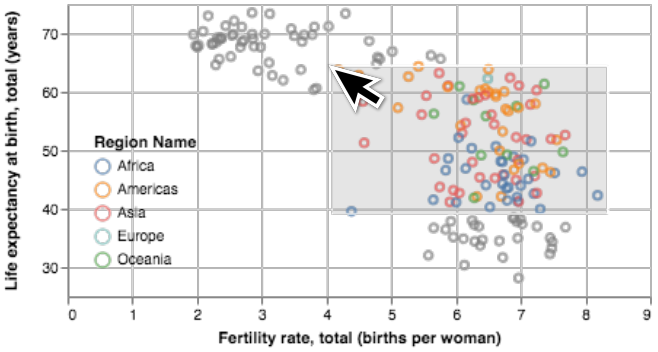
Grammar of Graphics



View Composition Algebra



Grammar of interaction

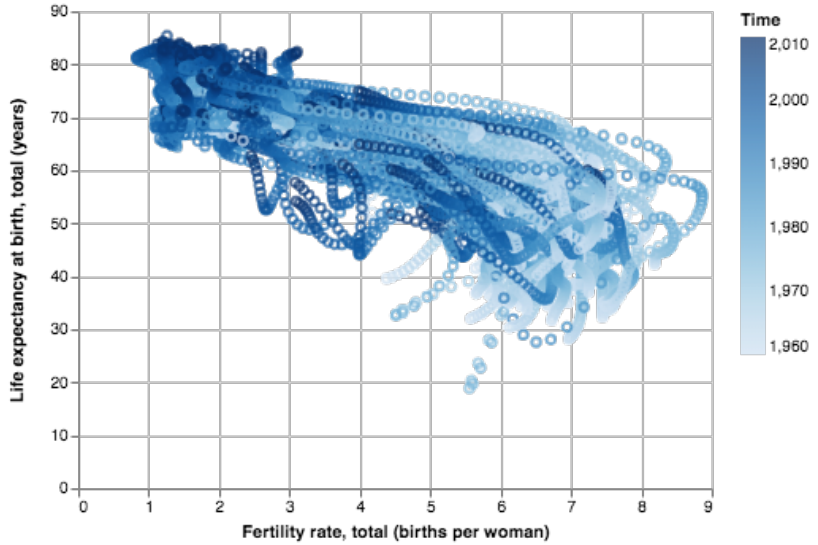


Vega-Lite presented by authors :

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

Grammar of Graphics

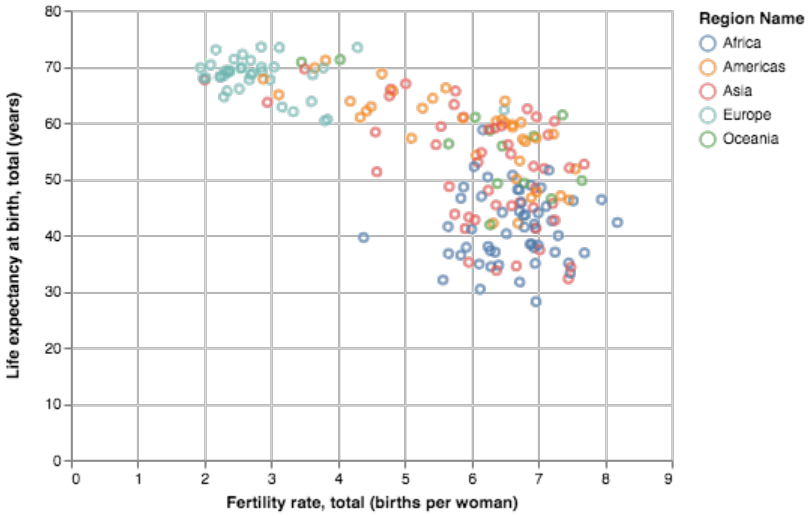
```
(
    alt.Chart( data = df_data_vis)
    .mark_point()
    .encode(
        x = 'Fertility rate',
        y = 'Life expectancy',
        color = 'Time'

    )
)
```



Data → Transformation → Encoding → Mark

Grammar of Graphics Basics

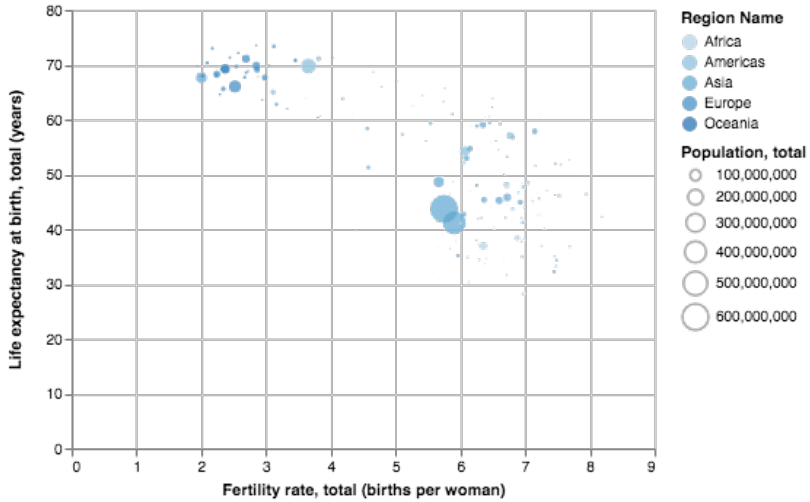
```
(
    alt.Chart( data = df_data_vis)
+   .transform_filter(alt.FieldEqualPredicate(1960,'Time'))
    .mark_point()
    .encode(
        x = 'Fertility rate',
        y = 'Life expectancy',
~   color = 'Region Name'

    )
)

```

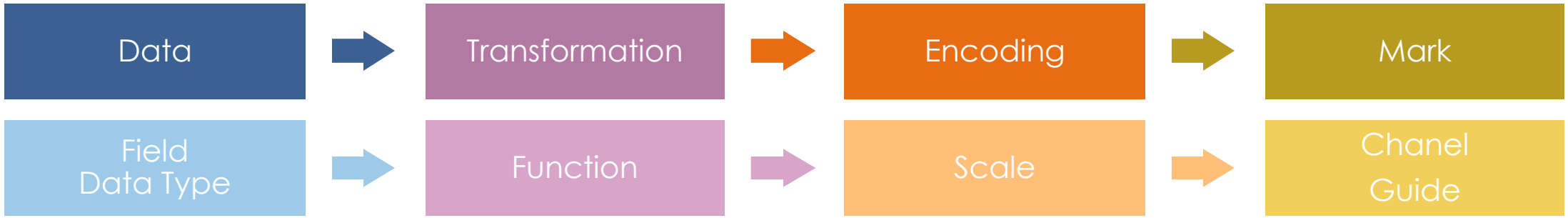


Data → Transformation → Encoding → Mark

Grammar of Graphics Encoding

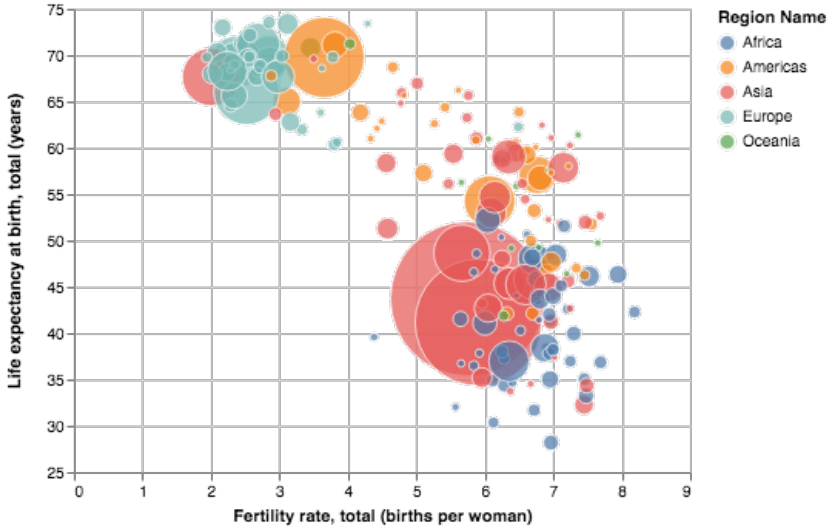
```
1  (
2      alt.Chart( data = df_data_vis)
3      .transform_filter(alt.FieldEqualPredicate(1960,'Time'))
4  ~   .mark_circle()
5      .encode(
6          x = 'Fertility rate',
7          y = 'Life ',
8  ~       color = 'Region Name:O',
9  +       size = 'Population',
10
11     )
12
13 )
```





Grammar of Graphics Scale

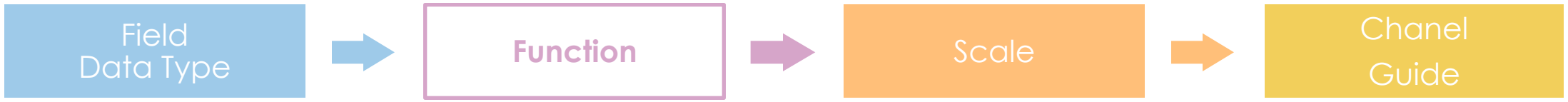
```
(
    alt.Chart( data = df_data_vis)
    .transform_filter(alt.FieldEqualPredicate(1960,'Time'))
    .mark_circle(stroke ='white',strokeWidth =1)
    .encode(
        x = alt.X('Fertility rate'),
        y = alt.Y(
            shorthand ='Life expectancy:Q',
            scale = alt.Scale(zero=False),
        ),
        size = alt.Size(
            field = 'Population',
            type = 'quantitative',
            legend = None,
            scale = alt.Scale(range=[20,10000])
        ),
        fill = alt.Fill('Region Name'),
    )
)
```



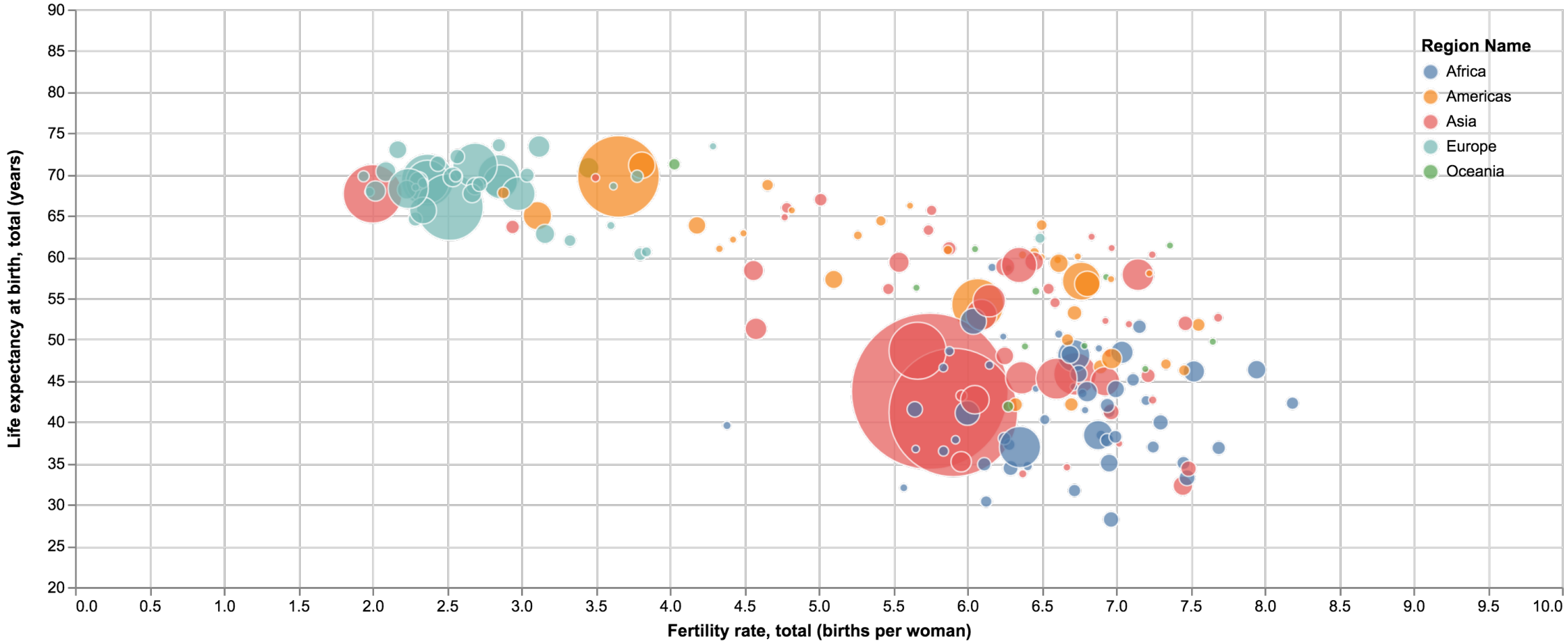
Field Data Type → Function → Scale → Chanel Guide

Grammar of Graphics Final

```
(
alt.Chart(data=df_data_vis, width=950, height=370)
.transform_filter(alt.FieldEqualPredicate(1960, "Time"))
.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 Name", legend=alt.Legend(orient="top-right")),
    alt.Tooltip("Country name"),
)
)
```

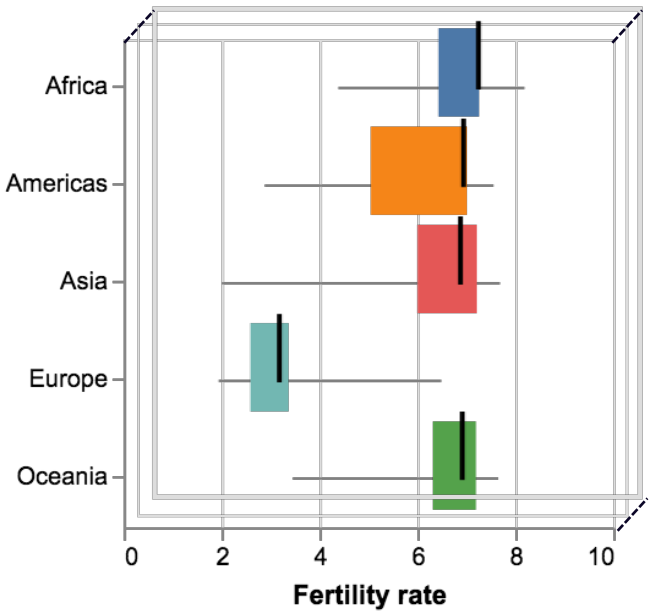


Basic static chart

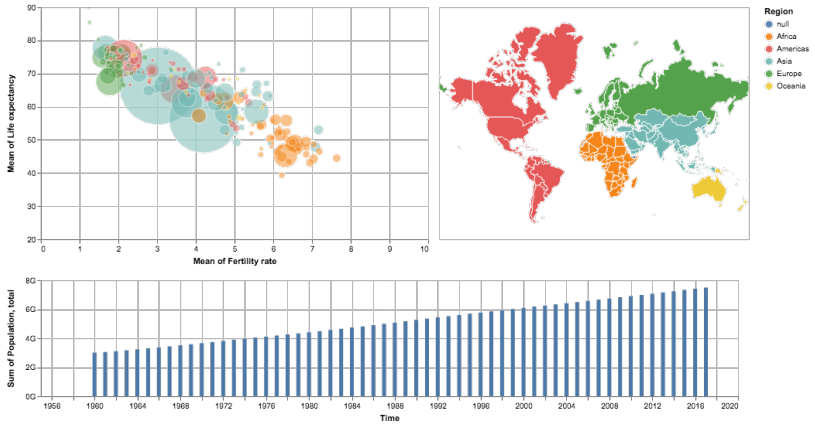


View Composition Algebra

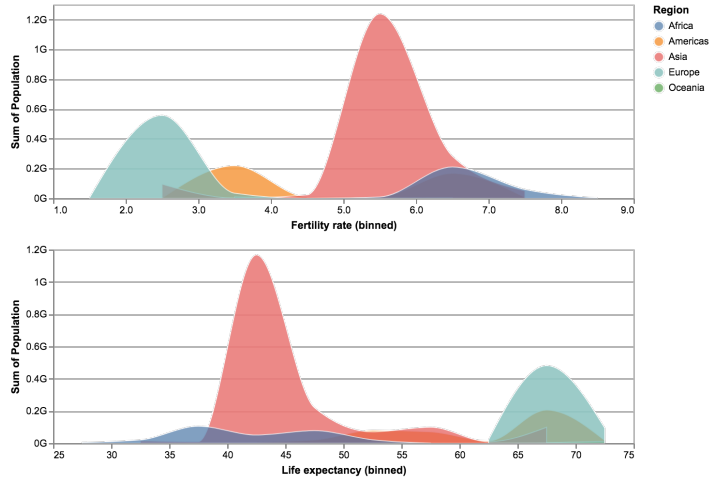
Layer



Concat



Facet/Repeat

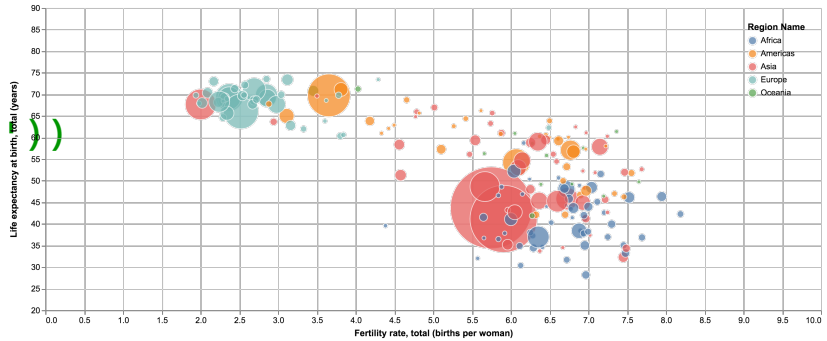


Resolve

Chanel Guide

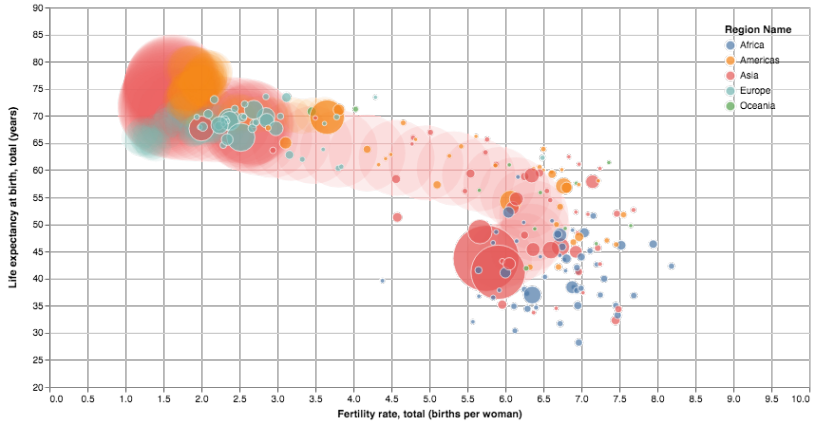
Layer: Prepare

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



Layer

```
+ bubles_history = (
+     base_bubles.mark_circle(strokeWidth=0, opacity=0.1)
+     .encode(alt.Fill("Region Name", legend=None))
+     .transform_filter(
+         alt.FieldOneOfPredicate(
+             field="Country name",
+             oneOf=["Russia", "US", "China"],
+         )
+     )
+ )
  bubles = base_bubles.transform_filter(
      alt.FieldEqualPredicate(1960, "Time")
  )
+ alt.layer(bubles_history, bubles)
```



TBD

Grammar of interaction

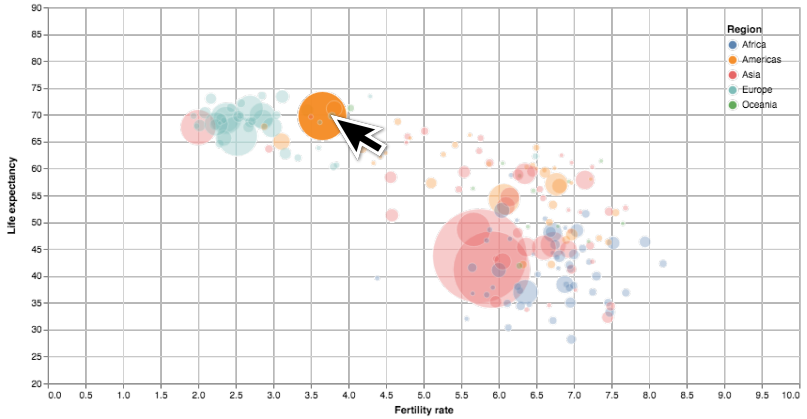
- **selection**
  - type
  - event
  - projection
  - binding
- condition
- predicat selected

Transformation

Condition

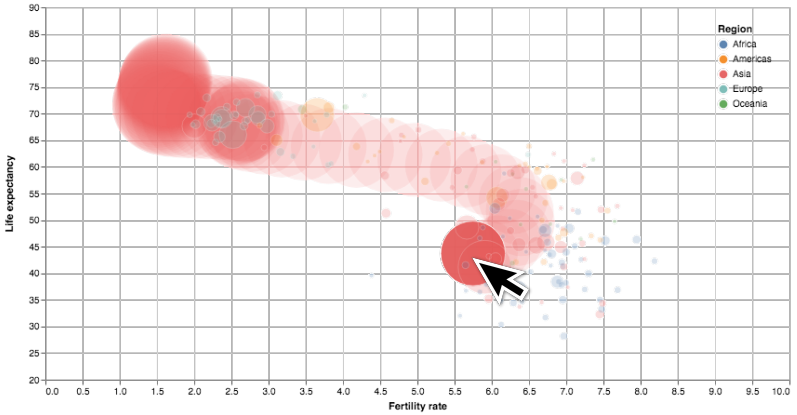
Selection

```
+ name_selection = alt.selection_multi(
+     on='mouseover', fields=['Country']
+ )
+ (
      base_bubles.transform_filter(
          alt.FieldEqualPredicate(1960, 'Year')
      )
+     .add_selection(name_selection)
+     .encode(
+         opacity=alt.condition(
+             name_selection, alt.value(0.9), alt.value(0.5)
+         )
+     )
+ )

```



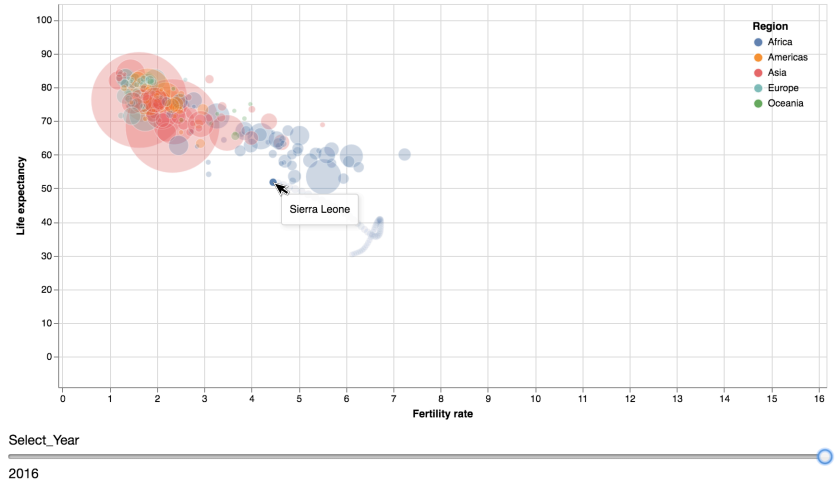
Layers with selection

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

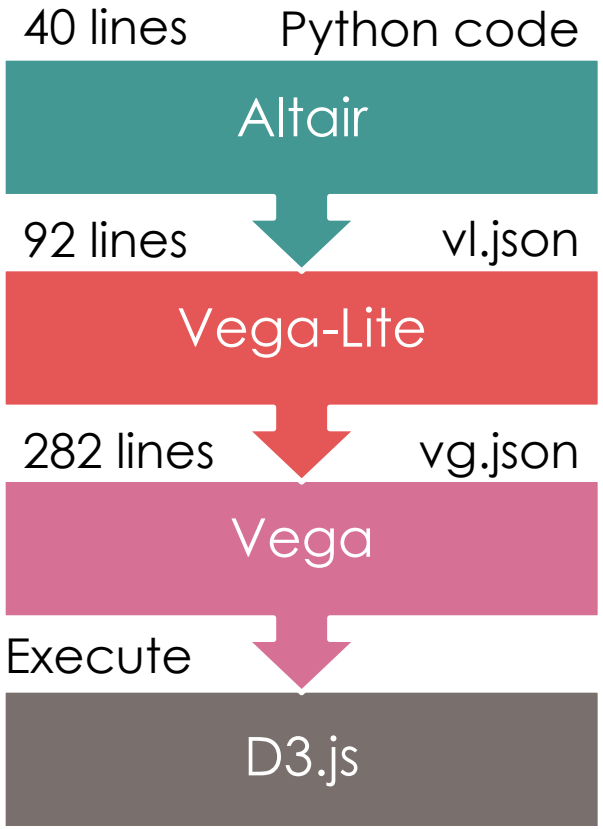


Binding selection

```
+ year_input = alt.binding_range(min=1960, max=2016, step=1)
+ year_selection = alt.selection_single(
+     bind=year_input, fields=['Year'], name="Select", empty='none'
+ )
  alt.layer(
      (
          base_bubles.mark_circle(strokeWidth=0, opacity=0.1)
          .encode(alt.Fill('Region', legend=None))
          .transform_filter(name_selection)
+         .add_selection(year_selection)
      ),
      (
~         base_bubles.transform_filter(year_selection)
          .add_selection(name_selection)
          .interactive()
          .encode(
              opacity=alt.condition(
                  name_selection, alt.value(0.9), alt.value(0.5)
              )
          )
      ),
  )
```



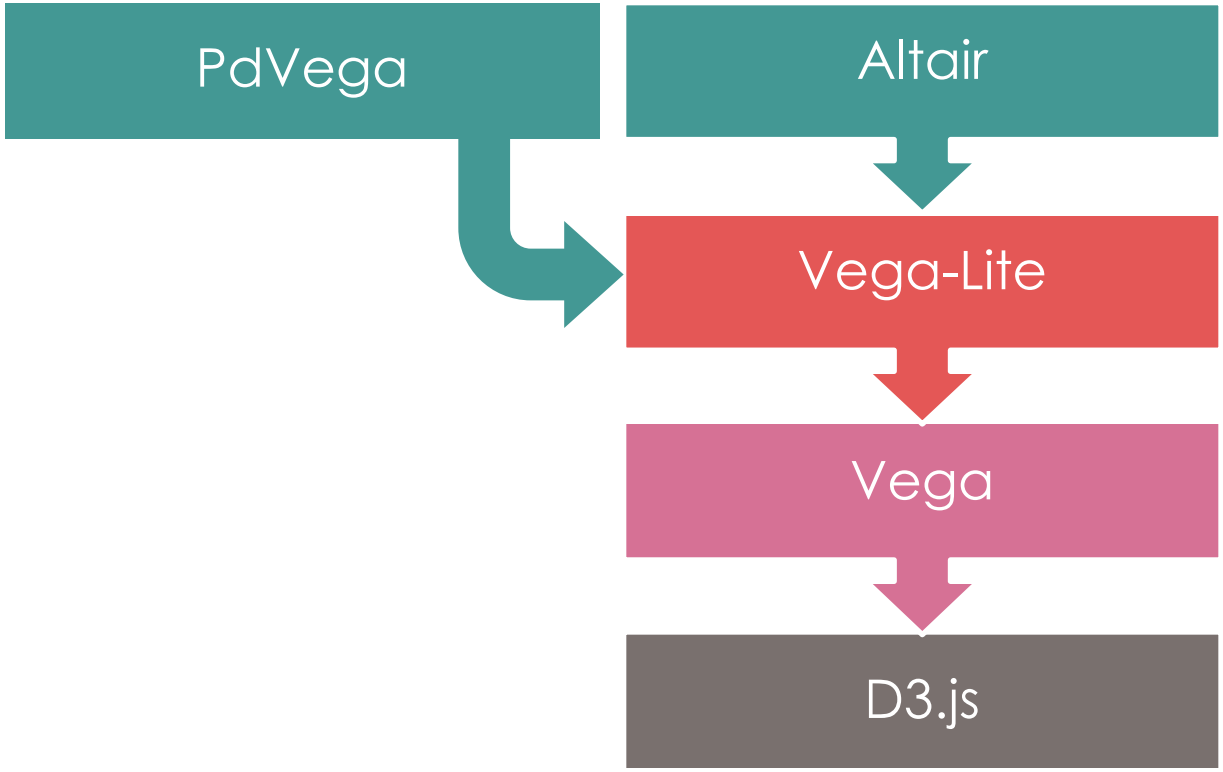
How it works?



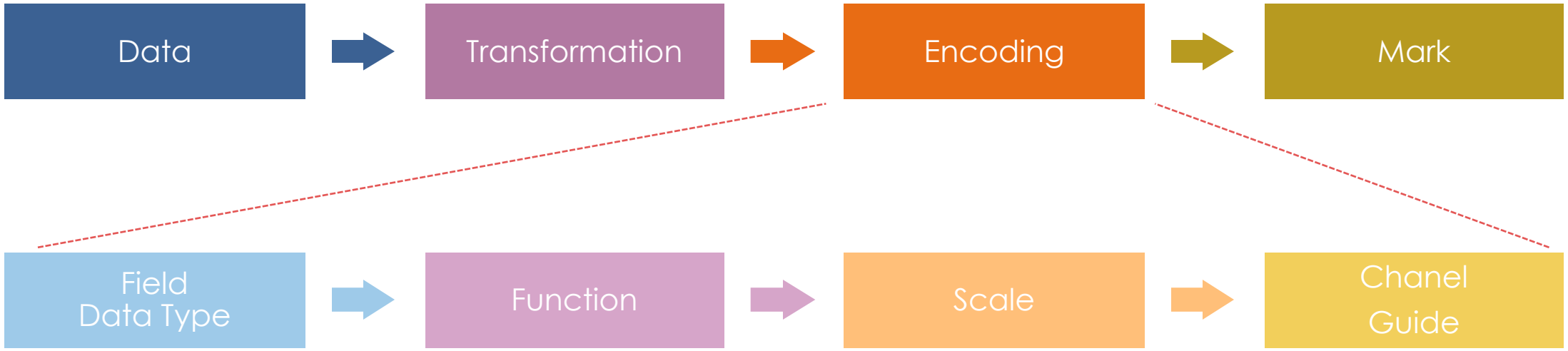
Vega-Lite specification (from first example)

```
{
"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