vegawidget

Composing and Rendering Interactive Vega(-Lite) Charts
vegawidget

Using Vega-Lite in the browser

Rendering Vega-Lite specifications as htmlwidgets

Composing Vega-Lite specifications
vegawidget

Ian Lyttle, Schneider Electric  @ijlyttle
Alicia Schep, Livongo  @AliciaSchep
Stuart Lee, Monash University  @_StuartLee
Haley Jeppson, Iowa State University  @heyhayhay__
Wenyu Yang, George Washington University  @iuysa1
Heike Hofmann, Iowa State University  @heike_hh
What is **vegawidget**?

- CRAN package to render Vega-Lite specs

- GitHub organization with packages to help build and work with Vega-Lite specs in R
Vega-Lite

Interactive grammar-of-graphics, rendered in the browser

Built on Vega: Vega-Lite is more concise, but less expressive

Developed by Interactive Data Lab, U Washington

Foundation for vegalite R package by Bob Rudis et al.
Vega-Lite: linked brushing
Vega-Lite: overview and detail
Grammar-of-Graphics as Food

salt fat acid heat

by Samin Nosrat

art by Wendy MacNaughton
ggplot2

data stat aes geom

Species
- setosa
- versicolor
- virginica

Petal.Length

Petal.Width
Vega-Lite

data transform encoding mark

Species
- setosa
- versicolor
- virginica

Petal.Length

Petal.Width
Vega-Lite

```json
{
  "data": {
    "values": [{"Petal.Width": 0.1, ...}, ...]
  },
  "mark": "point",
  "encoding": {
    "x": {
      "field": "Petal.Width",
      "type": "quantitative",
      "title": "Petal.Width"
    },
    "y": {
      "field": "Petal.Length",
      "type": "quantitative",
      "title": "Petal.Length"
    },
    "color": {
      "field": "Species",
      "type": "nominal"
    }
  }
}
```
Compose → Specification → Render → Vega-Lite

Icons made by Smashicons from www.flaticon.com
vegawidget

- htmlwidget
  - renders Vega(-Lite) specifications
  - provides access to interactivity
  - intended as low-level package
  - use vegawidget to render
  - use other packages to compose

https://vegawidget.github.io/vegawidget
list(
  data = list(values = iris),
  mark = "point",
  encoding = list(
    x = list(
      field = "Petal.Width",
      type = "quantitative",
      title = "Petal.Width"
    ),
    y = list(
      field = "Petal.Length",
      type = "quantitative",
      title = "Petal.Length"
    ),
    color = list(field = "Species", type = "nominal")
  )
) %>%
  as_vegaspec()
Interactivity

Vega gives access (via JS) to its:
- data
- signals (reactive variables)
- events

vegawidget gives access via Shiny

Shiny app by Stuart Lee
altair

- wraps Python Altair (via reticulate)
- by Jake VanderPlas, Brian Granger
- covers entire Vega-Lite API
- concise syntax
- https://altair-viz.github.io
- altair example-gallery
- reproduces entire Altair gallery

https://vegawidget.github.io/altair
alt$\text{Chart(iris)}$
mark_point()
encode(
  alt\$X(  
    "Petal Width:Q",
    title = "Petal.Width"
  ),
  alt\$Y(  
    "Petal Length:Q",
    title = "Petal.Length"
  ),
  color = "Species:N"
)$
properties(width = 300, height = 300)
vlbuildr

- API heavily inspired by vegalite (R)
- builds API semi-automatically based on the Vega-Lite schema
  - inspired by Altair (Python), vega-lite-api (JS)
- compose Vega-Lite specs using %>%

https://vegawidget.github.io/vlbuildr
vl_chart() %>%
  vl_add_data(values = iris) %>%
  vl_mark_point() %>%
  vl_encode_x(
    field = "Petal\Width:Q",
    title = "Petal.Width"
  ) %>%
  vl_encode_y(
    field = "Petal\Length:Q",
    title = "Petal.Length"
  ) %>%
  vl_encode_color("Species:N")
ggvega

- translate from **ggplot2** to Vega-Lite
- supported by Google Summer of Code
- obvs. inspired by **plotly**, Carson Sievert
- **ggplot2** & Vega-Lite are **declarative**
  - “what” not “how”
- ggvega translates only the declarations
- build & deploy Vega-Lite templates

https://vegawidget.github.io/ggvega
library("ggplot2")
library("ggvega")

gg_petal <- ggplot(iris) + geom_point(aes(x = Petal.Width, y = Petal.Length, colour = Species))

as_vegaspec(gg_petal)
library("ggplot2")
library("ggvega")
library("vlbuildr")

gg_petal <-
ggplot(iris) +
geom_point(
  aes(
    x = Petal.Width,
    y = Petal.Length,
    colour = Species
  )
)

as_vegaspec(gg_petal)
library("ggplot2")
library("ggvega")
library("vlbuildr")

gg_petal <- ggplot(iris) +
  geom_point(
    aes(
      x = Petal.Width,
      y = Petal.Length,
      colour = Species
    )
  )

as_vegaspec(gg_petal) %>%
  vl_encode_fill("Species:N")
Add interactivity

```r
# experiment: https://rstudio.cloud/project/398318

vl_petal <-
  as_vegaspec(gg_petal) %>%
  experimental_function_to_operate_on_layer({
    . %>%
    vl_encode_opacity(value = 0.3) %>%
    vl_add_interval_selection("brush") %>%
    vl_condition_opacity(
      selection = "brush",
      value = 1
    )
  })

# do the same with vl_sepal

# concatenate
vl_hconcat(vl_sepal, vl_petal)
```
Add interactivity
Summary

Compose → Specification → Render → Vega-Lite

Icons made by Smashicons from www.flaticon.com
Summary

Vega-Lite - interactive grammar-of-graphics
- JavaScript, rendered in the browser

vegawidget - htmlwidget, **render** Vega-Lite specifications
- provides interactive access to Vega-Lite charts

altair
vlbuildr - **compose** Vega-Lite specifications

ggvega
Composing and Rendering Interactive Vega(-Lite) Charts

<table>
<thead>
<tr>
<th>Vega-Lite (JS)</th>
<th><a href="https://vega.github.io/vega-lite">https://vega.github.io/vega-lite</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Altair (Python)</td>
<td><a href="https://altair-viz.github.io">https://altair-viz.github.io</a></td>
</tr>
</tbody>
</table>

vegawidget organization  https://github.com/vegawidget
📍 packages with pkgdown sites
https://vegawidget.rbind.io