

Interactivity

Session 10

PMAP 8921: Data Visualization with R
Andrew Young School of Policy Studies
Fall 2023

Plan for today

Making interactive graphics

Sharing content

Making interactive graphics

Three general methods

Single plots with {plotly}

Easy!

Dashboards with {flexdashboard}

Slightly more complicated

Complete interactive apps with Shiny

Super complicated!

Single plots with plotly

Plotly is special software for creating interactive plots with JavaScript

No knowledge of JavaScript needed!

`ggplotly()` in the `{plotly}` R package translates between R and Javascript for you!

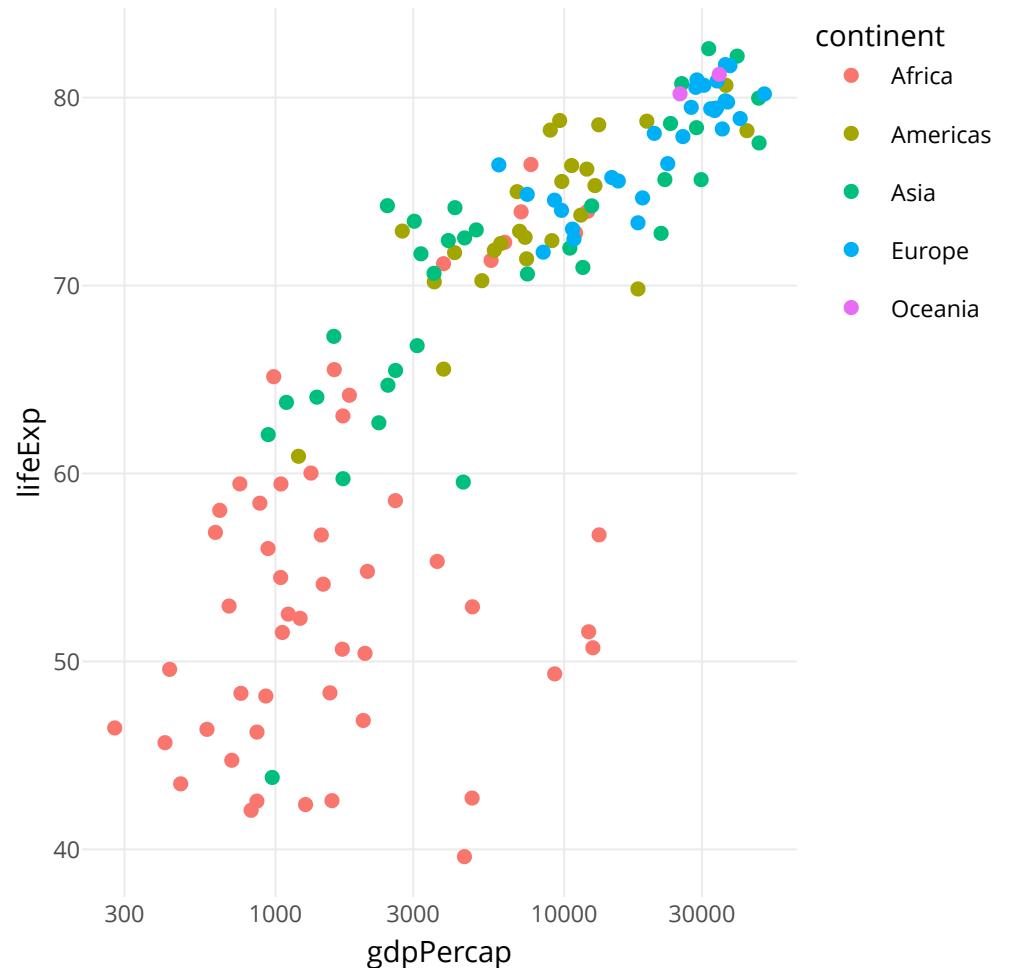
Plotly

```
library(gapminder)
library(plotly)

gapminder_2007 <- filter(gapminder,
                        year == 2007)

my_plot <- ggplot(
  data = gapminder_2007,
  mapping = aes(x = gdpPercap, y = lifeExp,
                color = continent)) +
  geom_point() +
  scale_x_log10() +
  theme_minimal()
```

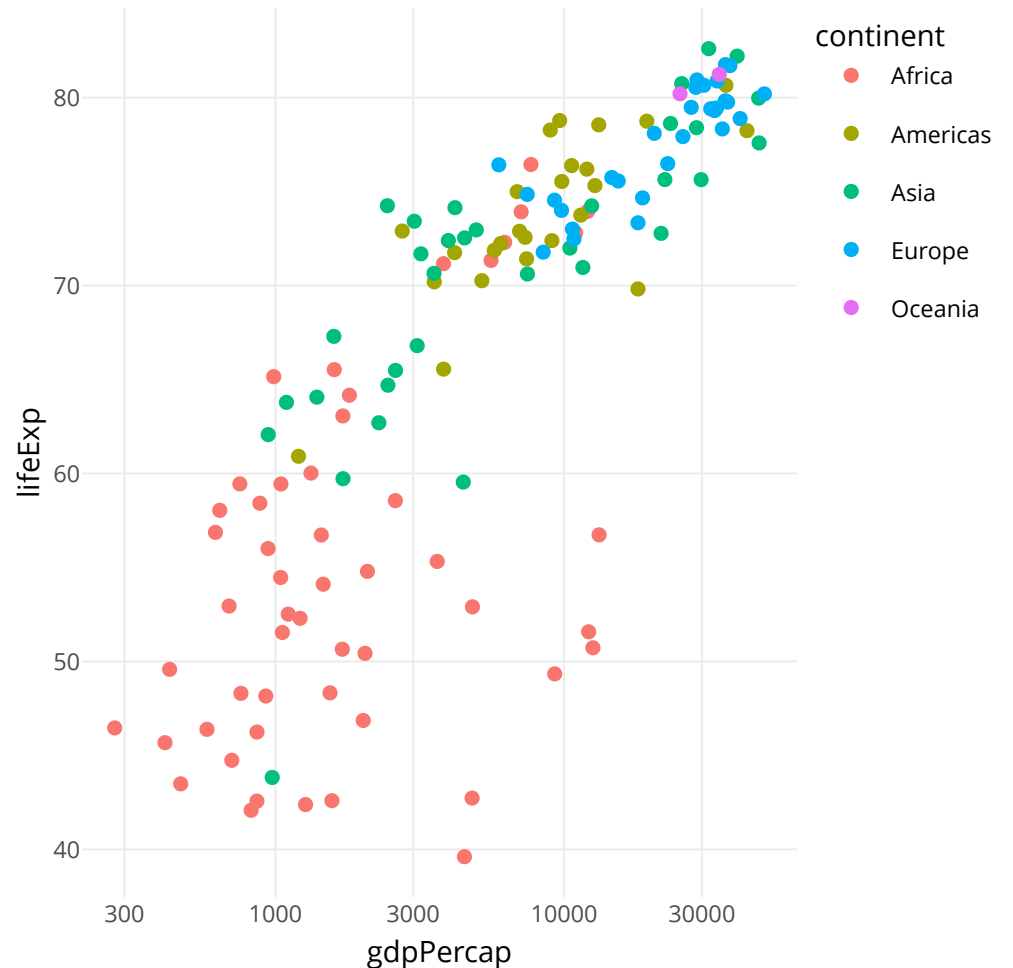
```
ggplotly(my_plot)
```



Plotly tooltips

```
my_plot <- ggplot(  
  data = gapminder_2007,  
  mapping = aes(x = gdpPercap, y = lifeExp,  
                color = continent)) +  
  geom_point(aes(text = country)) +  
  scale_x_log10() +  
  theme_minimal()
```

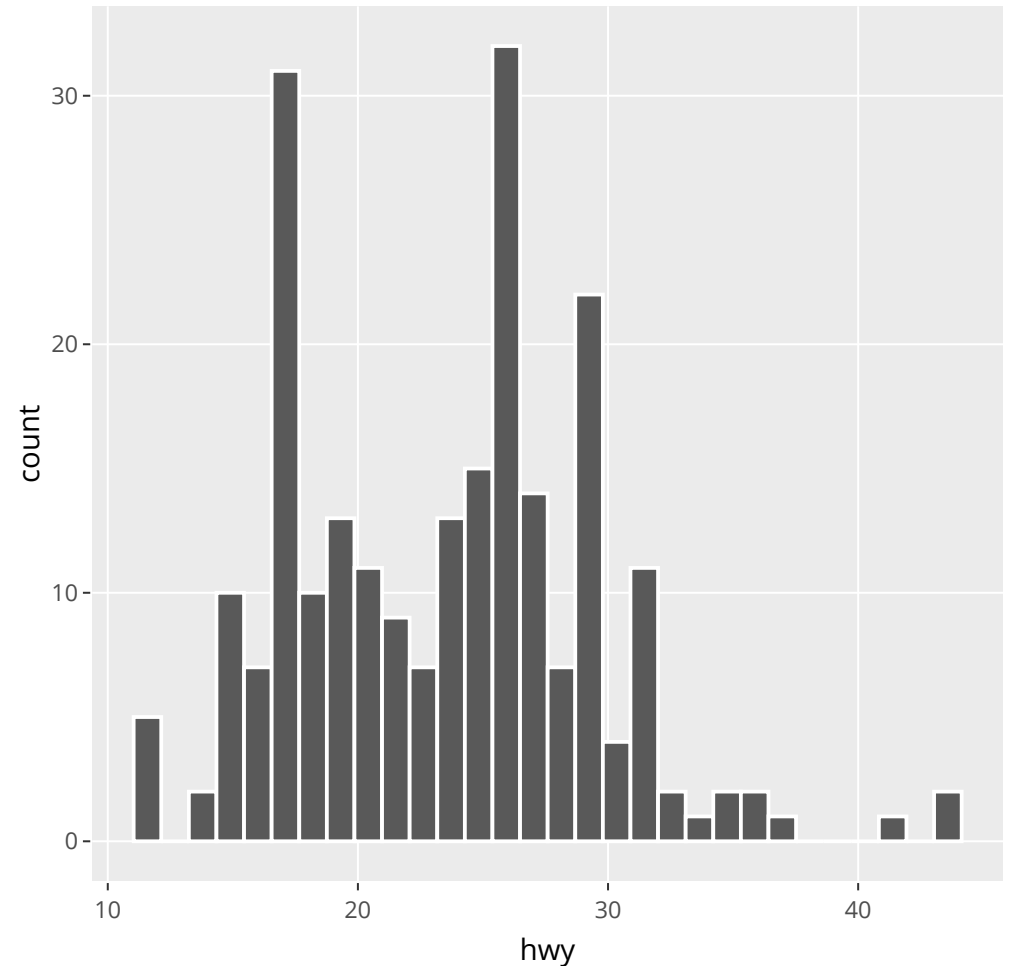
```
interactive_plot <- ggplotly(  
  my_plot, tooltip = "text"  
)  
interactive_plot
```



Works with most geoms!

```
car_hist <- ggplot(mpg,  
                  aes(x = hwy)) +  
  geom_histogram(binwidth = 2,  
                boundary = 0,  
                color = "white")
```

```
ggplotly(car_hist)
```



Save as HTML

Save a self-contained HTML version of it with `saveWidget()` in the `{htmlwidgets}` R package

```
# This is like ggsave, but for interactive HTML plots  
htmlwidgets::saveWidget(interactive_plot, "fancy_plot.html")
```

Fully documented

The **documentation** for ggplot2 + plotly is full of examples of how to customize everything

Rely on that ↑ + Google to make really fancy (and easy!) interactive plots

Three general methods

Single plots with {plotly}

Easy!

Dashboards with {flexdashboard}

Slightly more complicated

Dashboards with {flexdashboard}

Use basic R Markdown to build a dashboard!

```
1 |---
2 |title: "Single Column (Fill)"
3 |output:
4 |  flexdashboard::flex_dashboard:
5 |    vertical_layout: fill
6 |---
7 |
8 |### Chart 1
9 |
10| ```${r}```
11|
12| ```
13|
14|### Chart 2
15|
16| ```${r}```
17|
18| ```
19|
20|
21|
22|
23|
24|
25|
26|
```

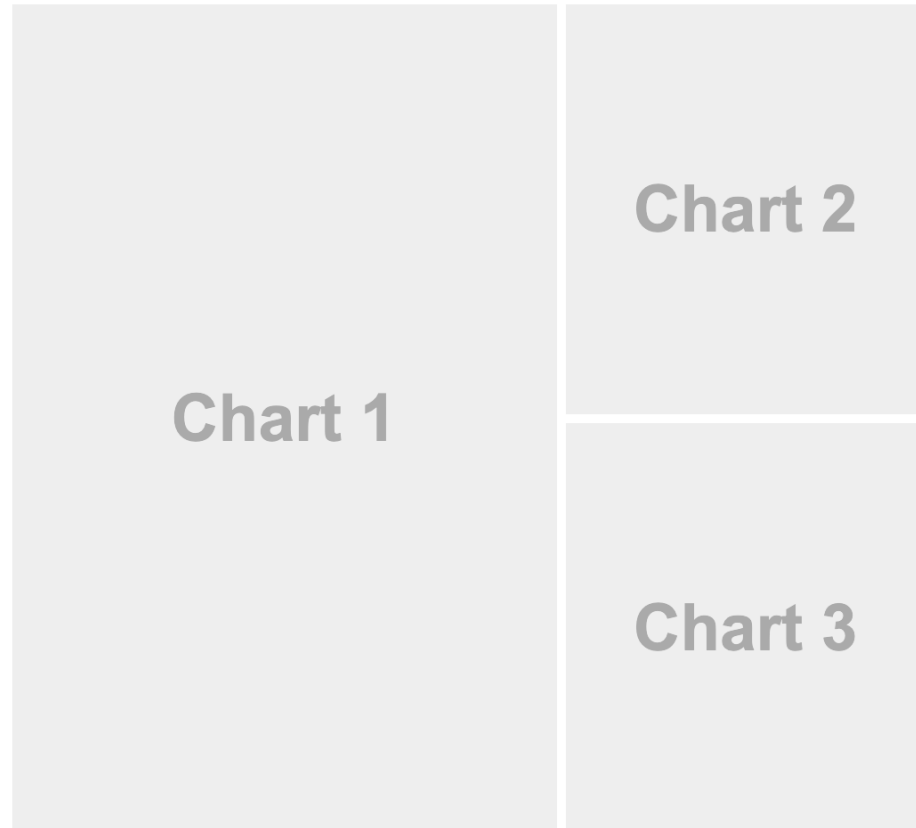
Chart 1

Chart 2

Dashboards with {flexdashboard}

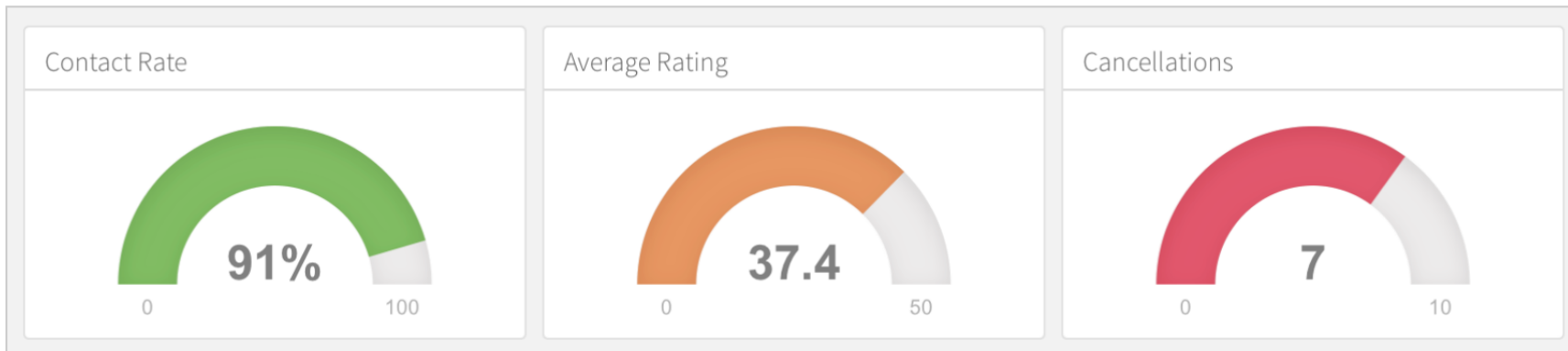
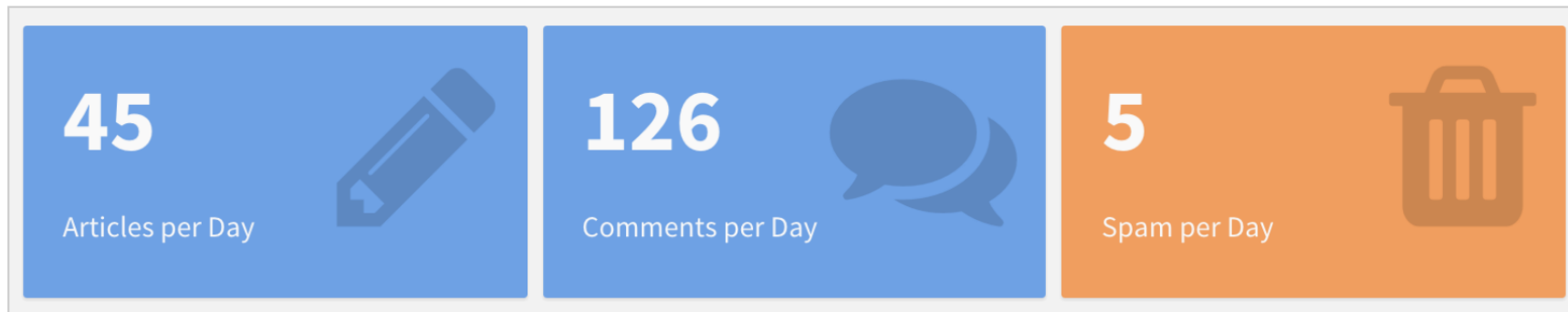
Make any kind of block arrangement

```
1 |---
2 |title: "Multiple Columns"
3 |output: flexdashboard::flex_dashboard
4 |---
5 |
6 |Column {data-width=600}
7 |-----
8 |
9 |### Chart 1
10|
11|```${r}```
12|
13|```
14|
15|Column {data-width=400}
16|-----
17|
18|### Chart 2
19|
20|```${r}```
21|
22|```
23|
24|### Chart 3
25|
26|```${r}```
27|
28|```
29|
```

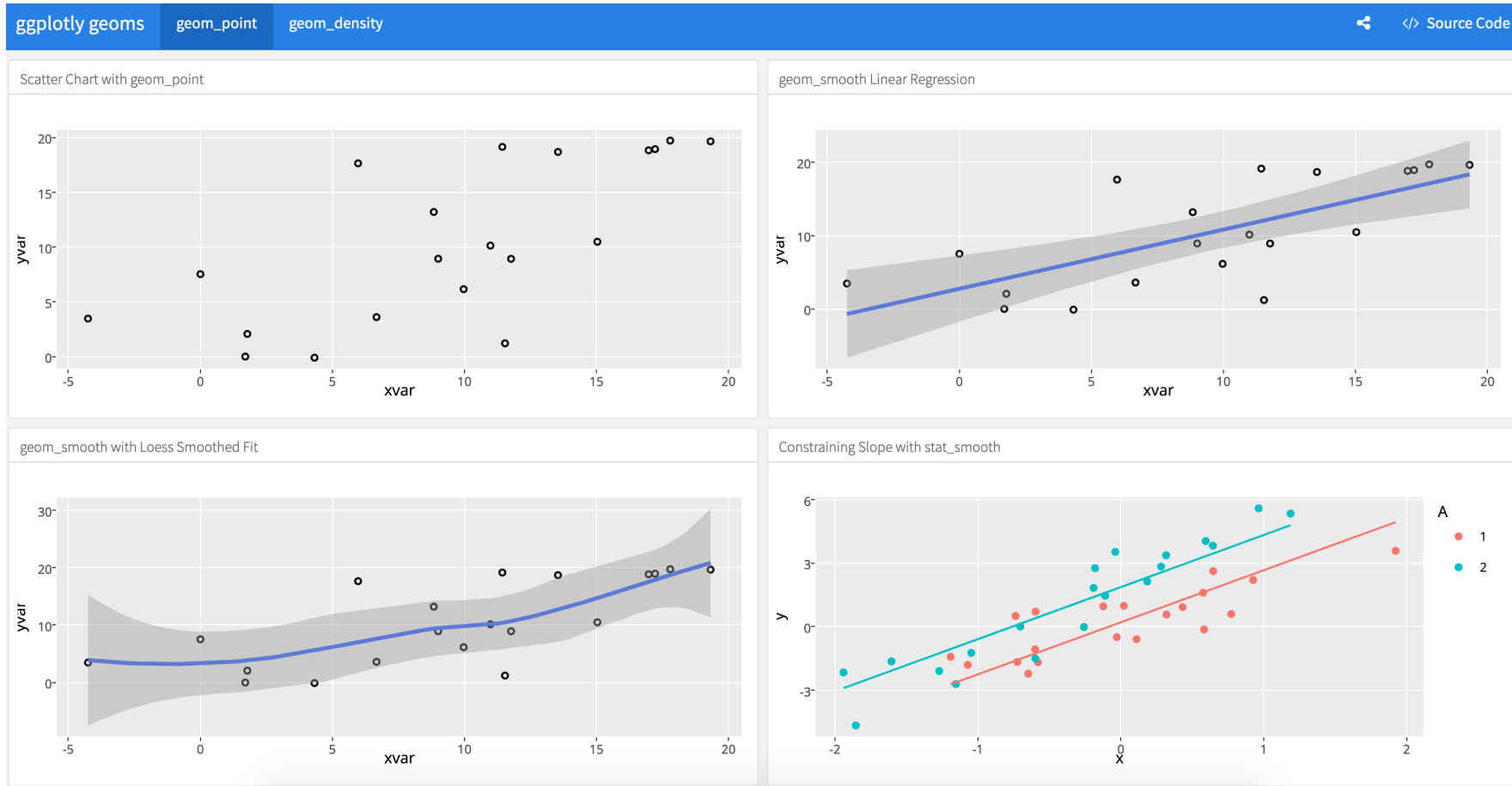


Dashboards with {flexdashboard}

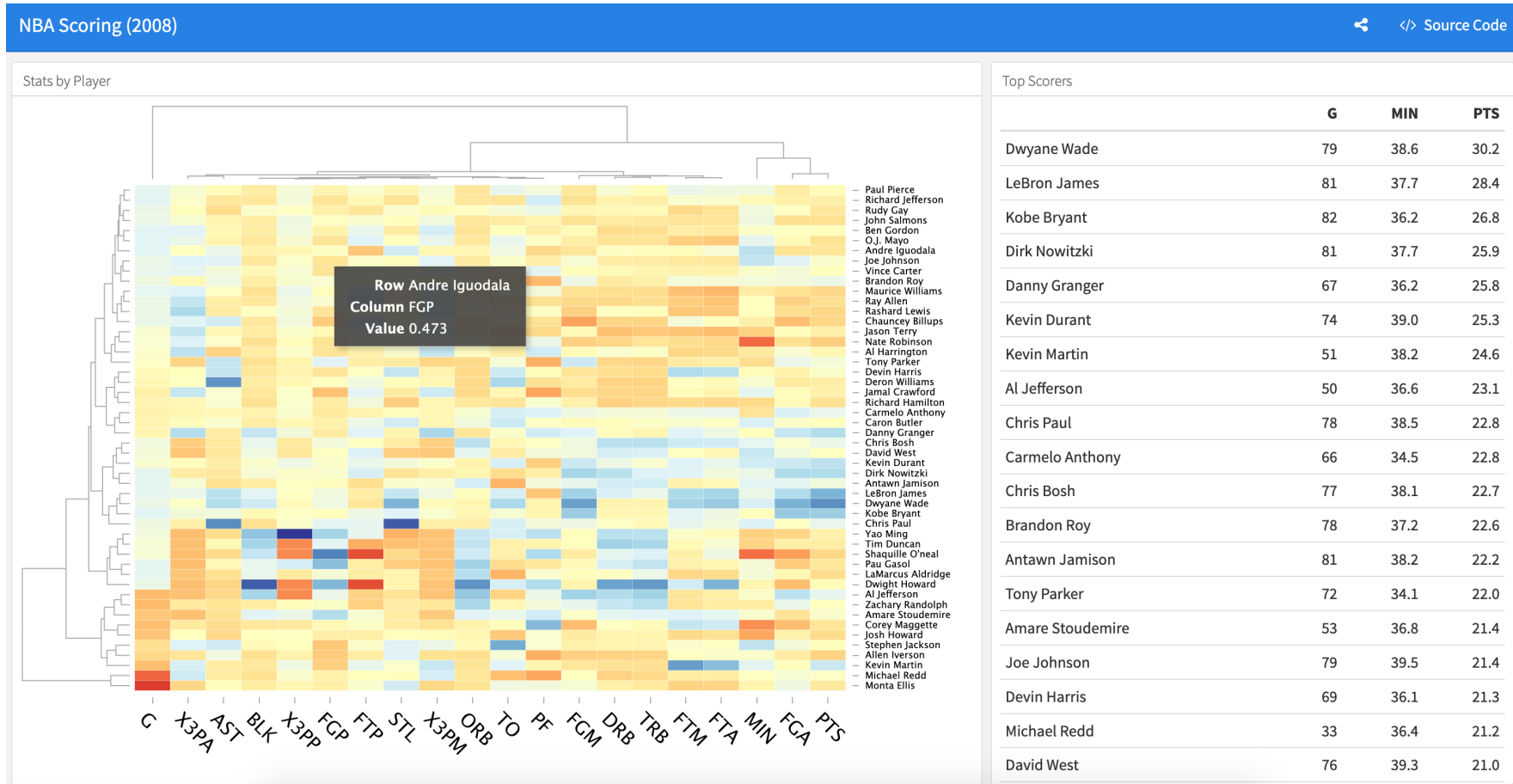
Add other elements like text and gauges



Example dashboards

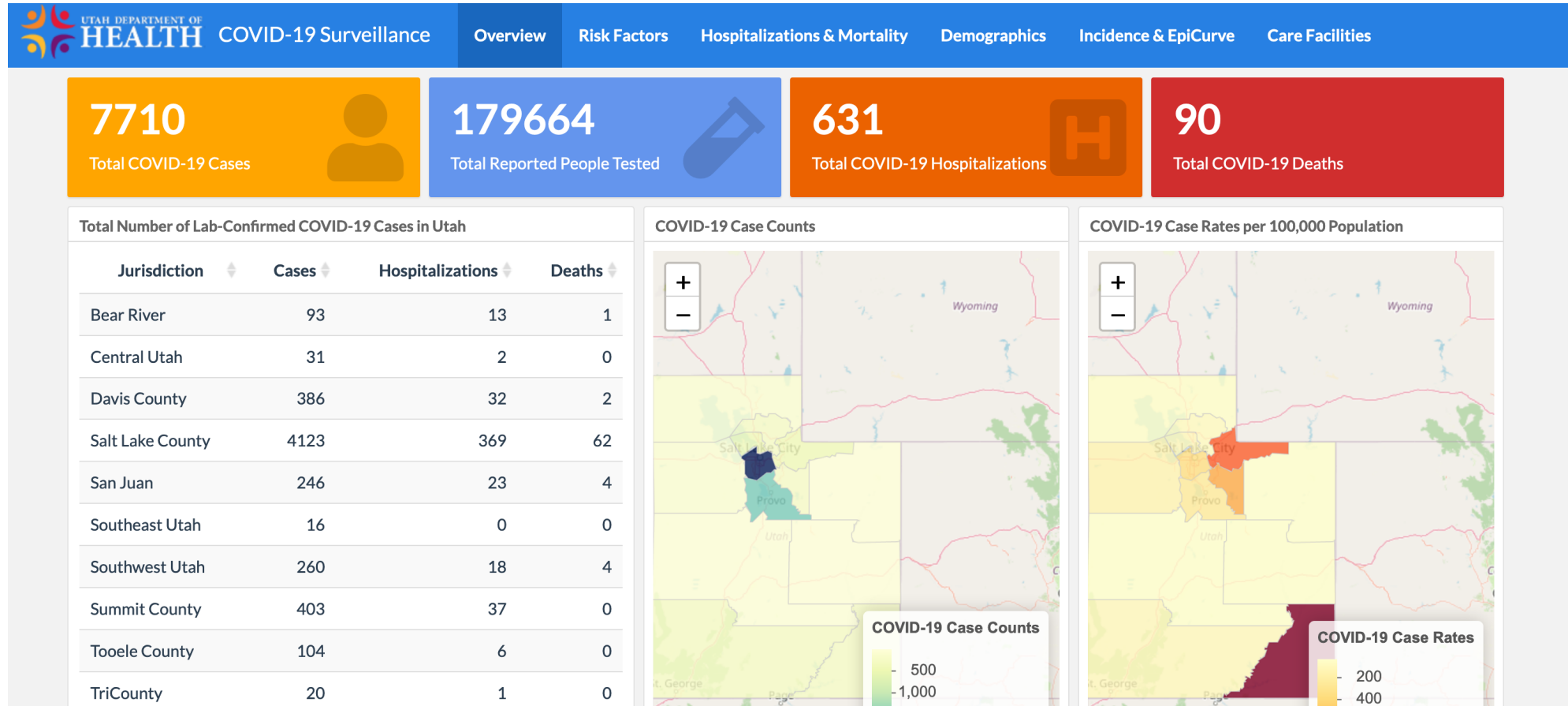


Example dashboards



NBA scoring

Example dashboards



Utah's COVID-19 dashboard

Outstanding documentation

The **documentation** for {flexdashboard} is full of examples and details of everything you can do

Rely on that ↑ + Google to make really fancy (and easy!) dashboards!

Three general methods

Single plots with {plotly}

Easy!

Dashboards with {flexdashboard}

Slightly more complicated

Complete interactive apps with Shiny

Super complicated!

Shiny

Shiny is a complete web application framework for interactive statistics

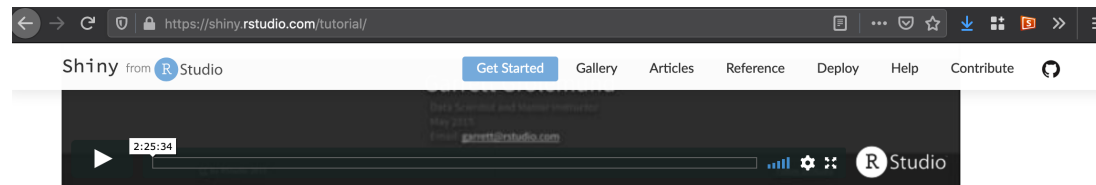
It's super complex and hard for beginners

I've never made a standalone Shiny app!

(And I don't plan on trying anytime soon)

Lots of resources to help start

RStudio has a whole website for helping you get started



Part 1 - How to build a Shiny app

1. [Introduction](#)
2. [R](#)
3. [App architecture](#)
4. [App template](#)
5. [Inputs and outputs](#)
6. [The server function](#)
7. [Sharing apps](#)
8. [Shinyapps.io](#)
9. [Shiny servers](#)
10. [Recap - Part 1](#)

Part 2 - How to customize reactions

11. [Introduction](#)
12. [Review of Part 1](#)
13. [Reactivity](#)
14. [Reactive values](#)
15. [Reactive functions](#)
16. [render*\(\)](#)
17. [reactive\(\)](#)
18. [isolate\(\)](#)
19. [observeEvent\(\)](#)
20. [eventReactive\(\)](#)
21. [reactiveValues\(\)](#)
22. [Recap - Part 2](#)
23. [Parting tips](#)

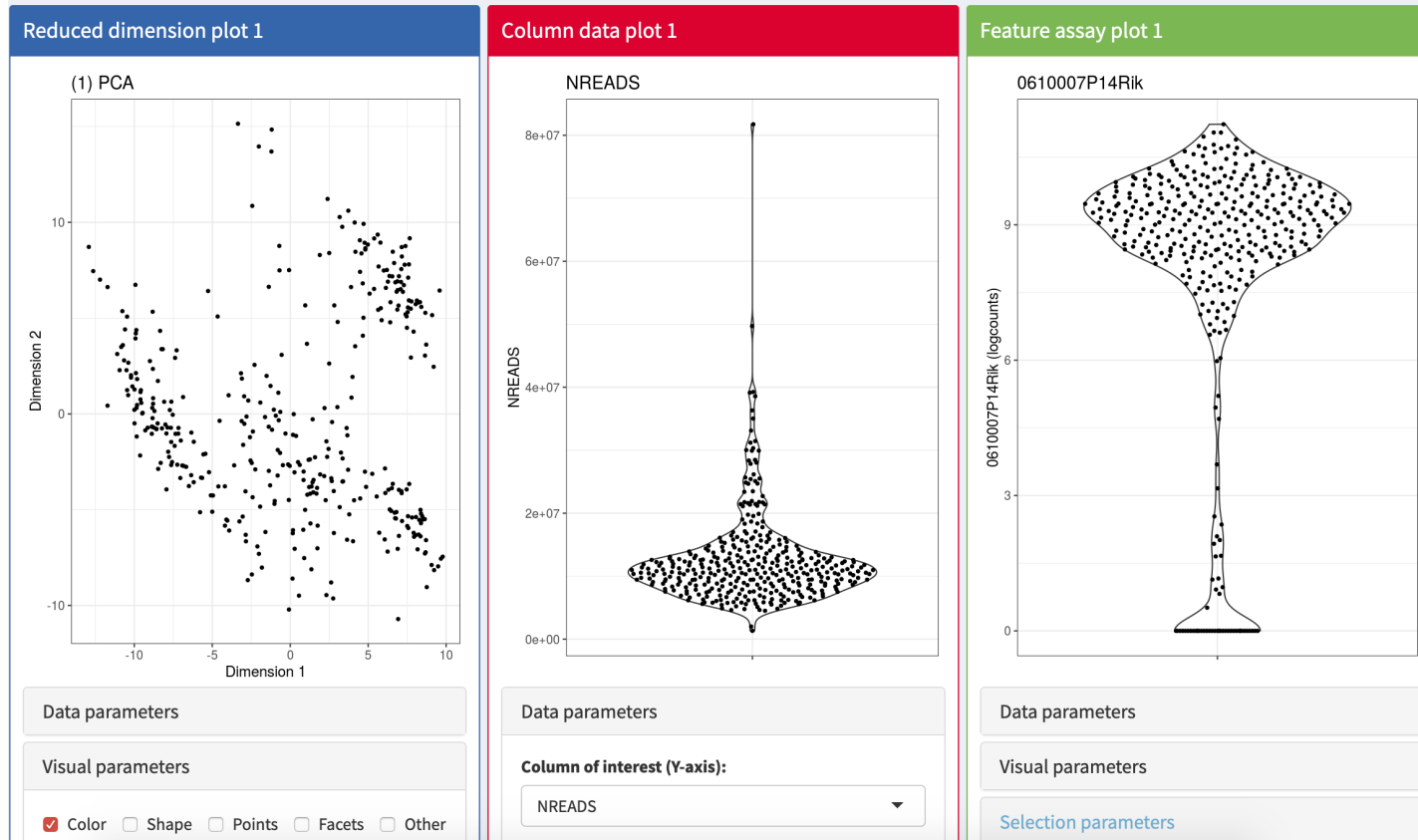
Part 3 - How to customize appearance

24. [Introduction](#)
25. [Review of Parts 1 and 2](#)
26. [HTML UI](#)
27. [Adding static content](#)
28. [Building layouts](#)
29. [Panels and tabsets](#)
30. [Prepackaged layouts](#)
31. [CSS](#)
32. [Recap - Part 3](#)

Getting started with Shiny

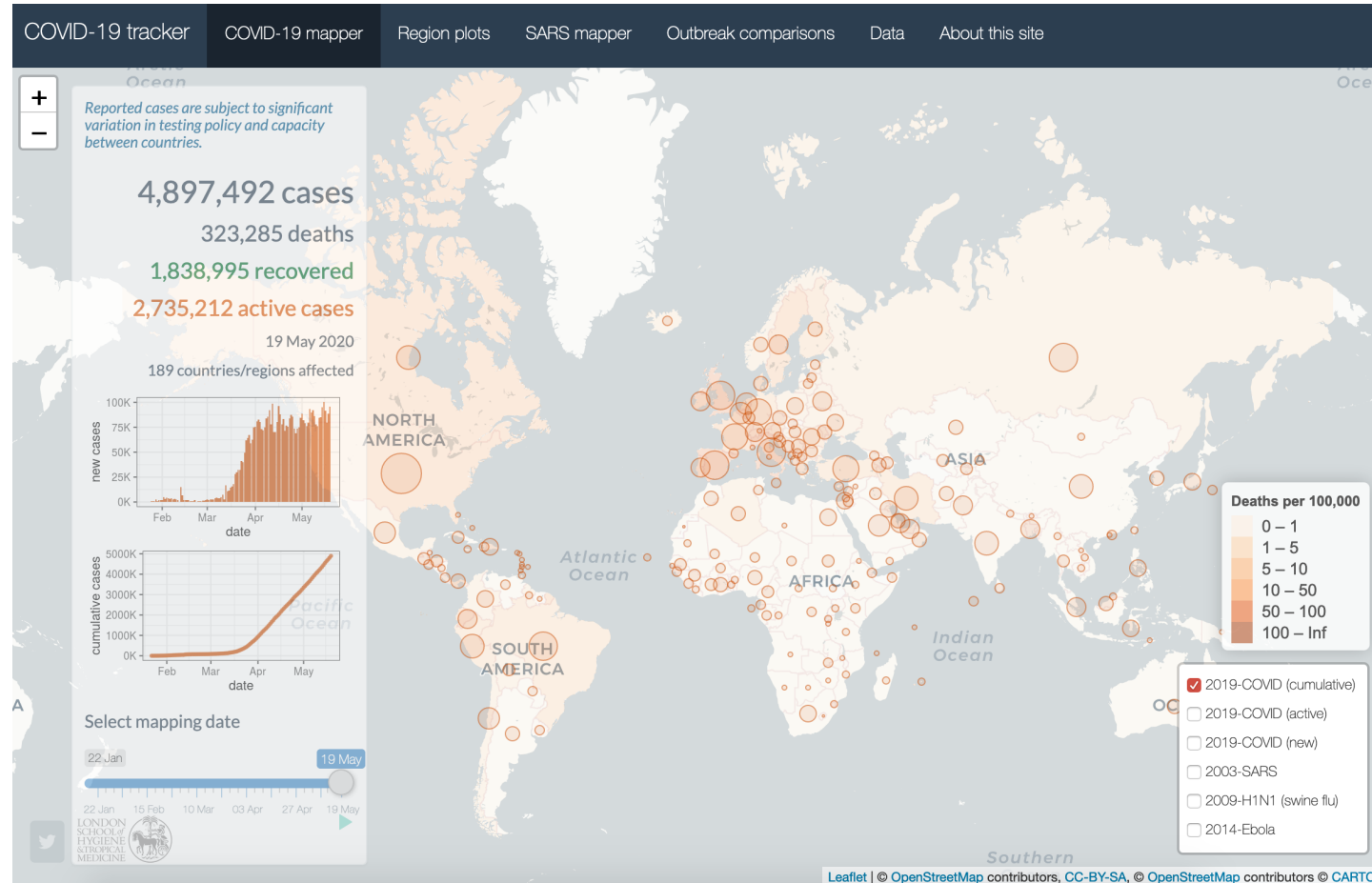
Really neat examples!

iSEE - interactive SummarizedExperiment Explorer v1.5.1



iSEE (interactive SummarizedExperiment Explorer)

Really neat examples!



COVID-19 tracker

Really neat examples!



Living in the Lego World

Demographics

Fashion

Moods

Ecology

About

Ethnicity and gender

Ethnic diversity and gender parity by theme

Find sets with a specific ethnicity or gender

Filter to one or more themes:

Nothing selected

Filter to one or more genders:

Nothing selected

Large graphs (e.g., of the full dataset) may take a few seconds to render. The first graph may take up to two minutes if the app is retrieving new data from Rebrickable.

Hover to see the part name.

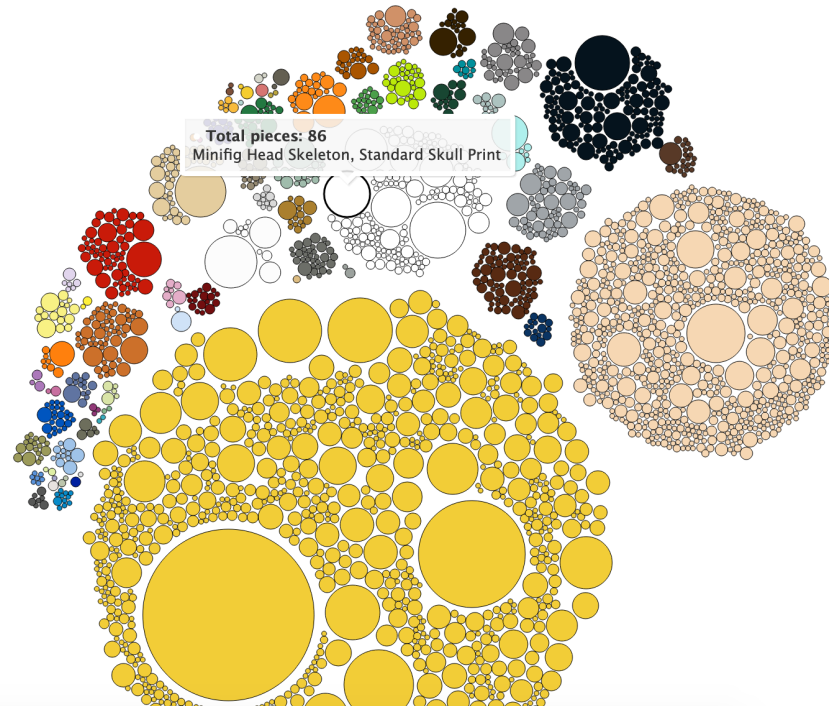
Each circle represents a **unique minifigure or minidoll head**.

Area is proportional to the **number of pieces** across all sets.

"**Ethnicity**" is the color of the piece. Yes, it's silly.

Gender is inferred from keywords in the part name ("Male", "Female", etc., plus references to facial hair).

Some heads are not labeled male/female but contain the name of a character of known gender (e.g., "Han Solo"). Incorporating this information would require a hand-maintained

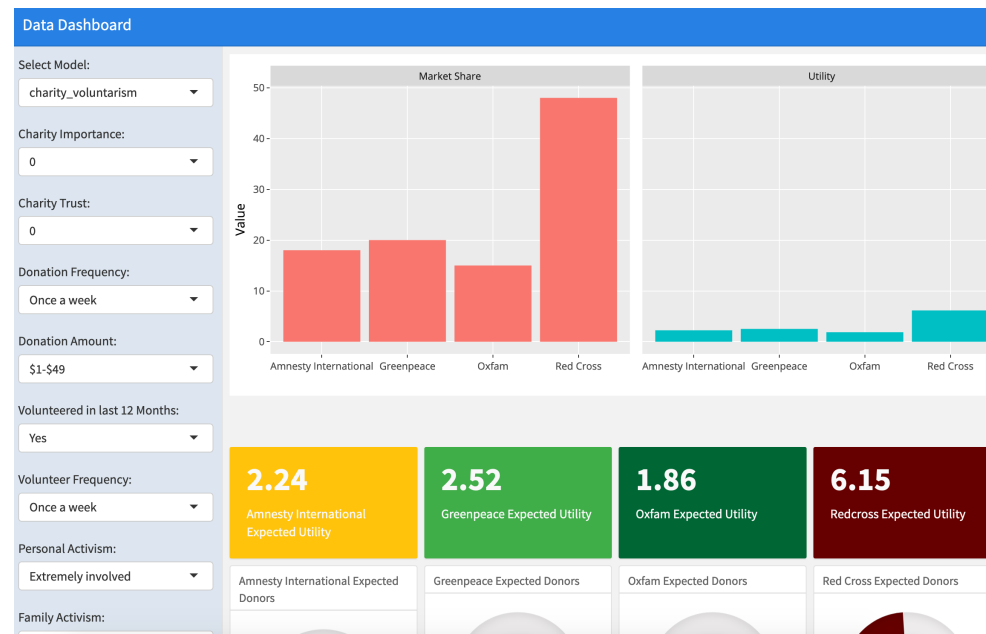


Living in the LEGO world

flexdashboard + Shiny

You can use reactive Shiny things in flexdashboards without building a complete Shiny app!

I have done this



Sharing content

What do you do after you knit?

When knitting to PDF or Word, you make a standalone file

E-mail it, message it, Slack it, whatever

When knitting to HTML, you make a website

By default it's a standalone `.html` file with graphics embedded, so you can still e-mail it, etc., but it can get huge if there are lots of images

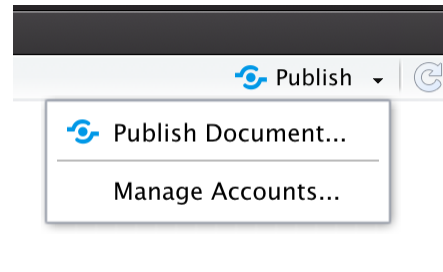
Standalone files won't work well if there's anything interactive

You can also post it online!

Places to put HTML documents

RPubs for knitted HTML documents

Built in to RStudio; works with ggplotly!



RPubs or **shinyapps.io** for flexdashboards

Your own web server for anything, if you have one

