

# DATA MATTERS.<sup>TM</sup>

## DATA SCIENCE SHORT COURSE SERIES

AUGUST 7 - 11, 2023

Welcome!  
We'll be  
starting soon!

VIRTUAL VIA ZOOM



ODUM INSTITUTE FOR  
RESEARCH IN SOCIAL SCIENCE

ncds

THE NATIONAL CONSORTIUM  
for DATA SCIENCE

renci

# Advanced Visualization in R: R Shiny

Angela Zoss

Data Matters, Fall 2023

<https://www.angelazoss.com/RShiny-1Day/>

# Slides and files

<https://github.com/amzoss/RShiny-1Day>

# Schedule

Session	Topics	Duration
Session 1	Introduction to data exploration, basic layout	9:30 a.m. – 10:35 a.m.
Morning break		10:35 a.m. – 10:50 a.m.
Session 2	Adding outputs, connecting basic inputs	10:50 a.m. – 11:55 a.m.
Lunch		11:55 a.m. – 1:10 p.m.
Session 3	Responding to user feedback, reactivity	1:10 p.m. – 2:15 p.m.
Afternoon break		2:15 p.m. – 2:30 p.m.
Session 4	Reducing duplication, using charts as inputs	2:30 p.m. – 3:35 p.m.
Q&A		3:35 p.m. – 3:40 p.m.

# Other course logistics

- We all have different skill levels here. That's great!
- Questions and interruptions are welcome, especially if you are lost. I want everyone to be able to follow along.
- You may know the answer to someone else's question. If it's quick, feel free to make suggestions in the Zoom chat. Otherwise, I'll be happy to address it myself.
- You may have more advanced suggestions on top of what I'm teaching. Please try not to share these in chat. Too much chat can be a distraction, and I have a specific sequence I follow to keep the content approachable. You can share advanced things in Slack instead.

# What is Shiny?

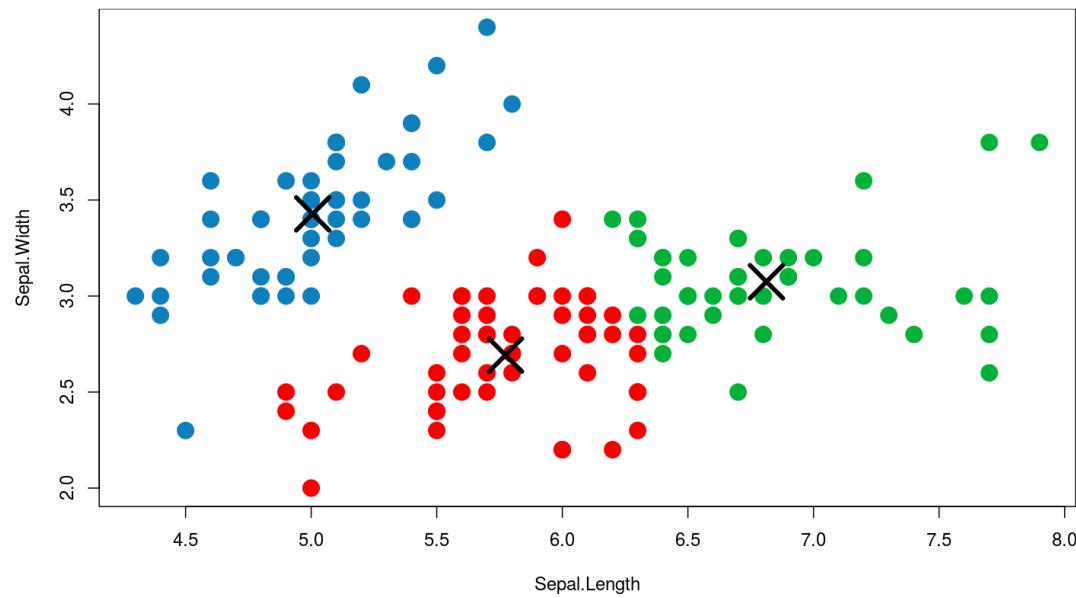
An interactive interface onto an R program

## Iris k-means clustering

X Variable

Y Variable

Cluster count



<https://shiny.posit.co/>

# Why make a web app in R?

- Generalized workflows
- Custom subsetting
- Changing parameters
- Personalizing output

The screenshot shows the Radiant web application interface. At the top, there's a navigation bar with tabs for Radiant, Data, Design, Basics, Model, Multivariate, Report, and various icons. Below the navigation bar, there are two main sections: 'Datasets' and 'Data preview'.

**Datasets:** A dropdown menu shows 'diamonds'. There are checkboxes for 'Add/edit data description' and 'Rename data'. Under 'Display:', a radio button is selected for 'preview' (the other options are 'str' and 'summary').

**Load data of type:** A dropdown menu shows 'rds | rda | rdata'. Below it is a 'Browse...' button and a message 'No file selected'.

**Save data to type:** A dropdown menu shows 'rds'. Below it is a 'Save' button with a download icon.

**Data preview:** This section shows a preview of the 'diamonds' dataset. The columns are labeled: price, carat, clarity, cut, color, depth, table, x, y, z, and date. The data table contains 10 rows of diamond attributes, such as a 580 carat Ideal cut diamond with a depth of 61.00 and a price of 580.

Below the preview table, a message says '10 of 3,000 rows shown. See View-tab for details.'

**Diamond prices**  
Prices of 3,000 round cut diamonds

**Description**  
A dataset containing the prices and other attributes of a sample of 3000 diamonds. The variables are as follows:

**Variables**

<https://shiny.posit.co/r/gallery/education/radiant/>

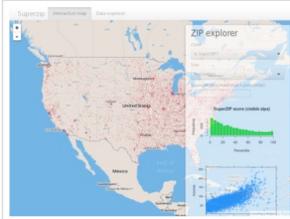
# How does Shiny compare to other tools?

- **Static reports**: easy to share and very stable, but no exploration
- **Crosstalk package in R**: easy to publish and has some simple options for responding to user input, but quite limited
- **Tableau**: easy to build dashboards and respond to user input, but not reproducible

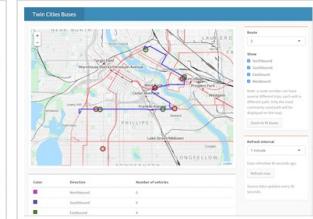
# Shiny examples

## Interactive visualizations

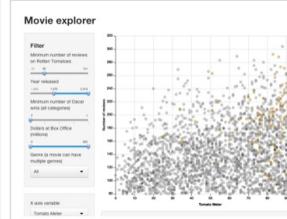
Shiny is designed for fully interactive visualization, using JavaScript libraries like [d3](#), [Leaflet](#), and [Google Charts](#).



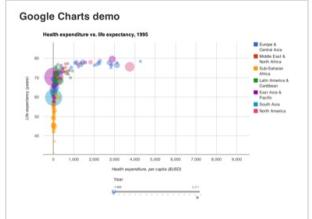
[SuperZip example](#)



[Bus dashboard](#)



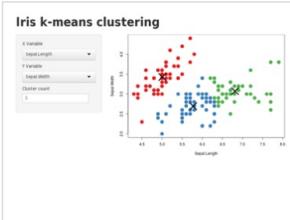
[Movie explorer](#)



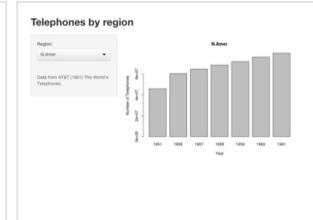
[Google Charts](#)

## Start simple

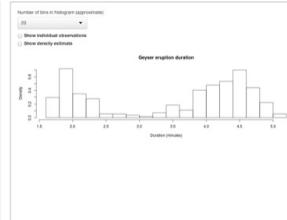
If you're new to Shiny, these simple but complete applications are designed for you to study.



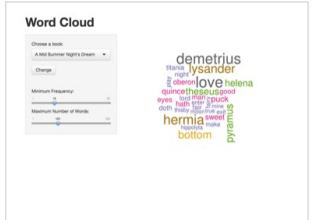
[Kmeans example](#)



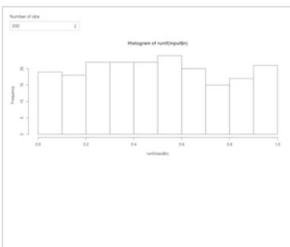
[Telephones by region](#)



[Faithful](#)



[Word cloud](#)



[Single-file shiny app](#)

<https://shiny.posit.co/r/gallery/>

Set up our environment

# Get workshop files

URL: <https://github.com/amzoss/RShiny-1Day>

## On GitHub:

- Click green “Code” button and select “Download ZIP”
- Unzip files on your computer
  - Windows: Double-click, then look for “Extract Files” at the top
  - Mac: Double-click
- Note: have noticed some issues when using OneDrive to store files

## In RStudio:

- Project → New project...
- Existing directory
- Select unzipped folder
- Create Project

# Shiny basics

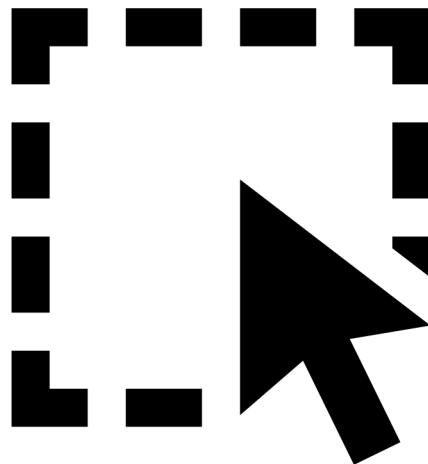
# Planning for Interactivity

## Output



Picking the right visual elements

## Input



Giving users the right controls

## Layout



Arranging everything in the right place

# How do you build a Shiny app?

- **Layout:** Setup the layout
- **Inputs and Outputs:** Insert visual components or placeholders
- **Server:** Write the R code to fill in the placeholders based on user feedback

# Shiny code is split into two main parts

## **User Interface (UI)**

the website people will see and interact with

## **Server**

special R code that generates and updates content for the website

# Running the app on your computer

- Develop the app in RStudio
- Click Run App
- View in RStudio or in browser

# Sharing an app

- Shiny Apps (publish a small number of apps for free, pay for more)

<http://www.shinyapps.io/>

- Shiny Server (host your own server for free)

<https://posit.co/download/shiny-server/>

- Posit Connect (enterprise solution, high fees)

<https://posit.co/products/open-source/shinyserver/>

# Fixing Errors

# Debugging code

- Start simple
  - If you see an error:
    - read error message for hints
    - check for problems with spelling/punctuation marks
  - Get code to run without errors
  - Check result to see if it makes sense
- 
- Add a small change
  - Get code to run without errors
  - Check result to see if it makes sense
  - etc.

# RStudio built-in help documentation

- In console, type  
?<function or package name>
- In help tab (not help menu), type into main search box
- In package tab, click on package name
- In help menu, use the Cheat Sheets submenu to download cheat sheet PDFs

# Shiny Cheat Sheet

Help →  
Cheatsheets →  
Web Applications with shiny

The Shiny Cheatsheet is a comprehensive guide for creating web applications in R. It includes sections for building an app, sharing your app, rendering outputs, and collecting inputs.

### Building an App

A Shiny app is a web page (UI) connected to a computer running a live R session (server). In ui nest R functions to build an HTML interface. To generate the template, type `shinyapp` and press Tab in the RStudio IDE or go to File > New Project > New Directory > Shiny Application. The UI code shows how to add inputs with `*input()` functions and outputs with `*output()` functions. The server code shows how to render outputs and respond to inputs with R. Save your template as `app.R`. Keep your app in a directory, which will cause the server to update the UI's displays (by running R code). Tell the server how to render outputs and respond to inputs with R. Refer to UI inputs with `input$<id>` and outputs with `output$<id>`. Call `shinyApp()` to combine ui and server into an interactive app!

### Share

Share your app in three ways:

- Host it on [shinyapps.io](#), a cloud based service from Posit. To deploy Shiny apps:
  - Create a free or professional account at [shinyapps.io](#)
  - Click the Publish icon in RStudio IDE, or run: `rconnect::deployApp(<path to directory>)`
- Purchase Posit Connect, a publishing platform for R and Python. [posit.co/products/enterprise/connect/](#)
- Build your own Shiny Server [posit.co/products/open-source/shinyserver/](#)

### Outputs

`render()` and `*output()` functions work together to add R output to the UI.

Output Type	Description
<code>dataTableOutput(outputId)</code>	DT, <code>renderDataTable(expr, options, searchDelay, callback, escape, env, quoted, outputArgs)</code>
<code>imageOutput(outputId, width, height, click, dblclick, hover, brush, inline)</code>	<code>renderImage(expr, env, quoted, deleteFile, outputArgs)</code>
<code>plotOutput(outputId, width, height, click, dblclick, hover, brush, inline)</code>	<code>renderPlot(expr, width, height, res, ..., alt, env, quoted, execOnResize, outputArgs)</code>
<code>verbatimTextOutput(outputId, placeholder)</code>	<code>renderPrint(expr, striped, hover, bordered, spacing, width, align, rownames, colnames, digits, na, ..., env, quoted, outputArgs)</code>
<code>tableOutput(outputId)</code>	<code>renderTable(expr, env, quoted, outputArgs, sep)</code>
<code>textOutput(outputId, container, inline)</code>	<code>renderText(expr, env, quoted, outputArgs)</code>
<code>uiOutput(outputId, inline, container, ...)</code>	<code>renderUI(expr, env, quoted, outputArgs)</code>
<code>htmlOutput(outputId, inline, container, ...)</code>	<code>uiOutput(outputId, inline, container, ...)</code>

These are the core output types. See [htmlwidgets.org](#) for many more options.

### Inputs

Collect values from the user. Access the current value of an input object with `input$<inputId>`. Input values are reactive.

Input Type	Description
<code>actionButton(inputId, label, icon, width, ...)</code>	
<code>actionLink(inputId, label, icon, ...)</code>	
<code>checkboxGroupInput(inputId, label, choices, selected, inline, width, choiceNames, choiceValues)</code>	<code>checkboxGroupInput(inputId, label, choices, selected, inline, width, choiceNames, choiceValues)</code>
<code>checkboxInput(inputId, label, width)</code>	<code>checkboxInput(inputId, label, width)</code>
<code>dateInput(inputId, label, value, min, max, format, startView, weekstart, language, width, autoclose, datesDisabled, daysOfWeekDisabled)</code>	<code>dateInput(inputId, label, value, min, max, format, startView, weekstart, language, separator, width, autoclose)</code>
<code>dateRangeInput(inputId, label, start, end, min, max, format, startView, weekstart, language, separator, width, autoclose)</code>	<code>dateRangeInput(inputId, label, start, end, min, max, format, startView, weekstart, language, separator, width, autoclose)</code>
<code>fileInput(inputId, label, multiple, accept, width, buttonLabel, placeholder)</code>	<code>fileInput(inputId, label, multiple, accept, width, buttonLabel, placeholder)</code>
<code>numericInput(inputId, label, value, min, max, step, width)</code>	<code>numericInput(inputId, label, value, min, max, step, width)</code>
<code>passwordInput(inputId, label, value, width, placeholder)</code>	<code>passwordInput(inputId, label, value, width, placeholder)</code>
<code>radioButtons(inputId, label, choices, selected, inline, width, choiceNames, choiceValues)</code>	<code>radioButtons(inputId, label, choices, selected, inline, width, choiceNames, choiceValues)</code>
<code>selectInput(inputId, label, choices, selected, multiple, selectize, width, size)</code>	<code>selectInput(inputId, label, choices, selected, multiple, selectize, width, size)</code>
<code>sliderInput(inputId, label, min, max, value, round, format, locale, ticks, animate, width, sep, pr, pg, timeFormat, timeRange)</code>	<code>sliderInput(inputId, label, min, max, value, round, format, locale, ticks, animate, width, sep, pr, pg, timeFormat, timeRange)</code>
<code>textInput(inputId, label, value, width, placeholder)</code>	<code>textInput(inputId, label, value, width, placeholder)</code>
<code>textAreaInput()</code>	<code>textAreaInput()</code>

# Basic Layout

# Mechanics of Shiny Layouts

- Build a page out of containers, which are all R functions
- Insert visual components into those functions as a list separated by commas

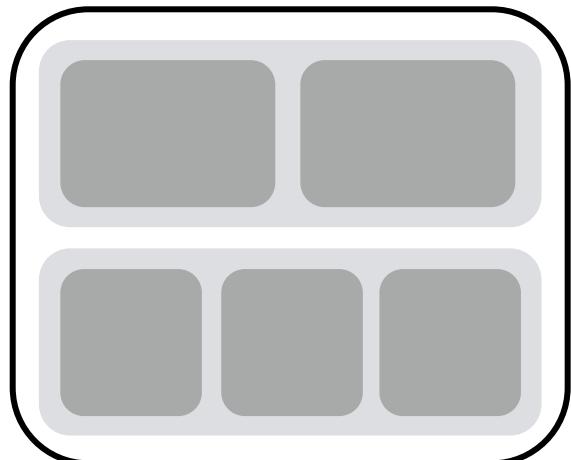
Examples:

```
layoutContainer(visualComponent(...), visualComponent2(...))  
layoutContainer(  
    visualComponent(...),  
    layoutContainer(visualComponent2(...), visualComponent3(...))  
)
```

# Start with a Page

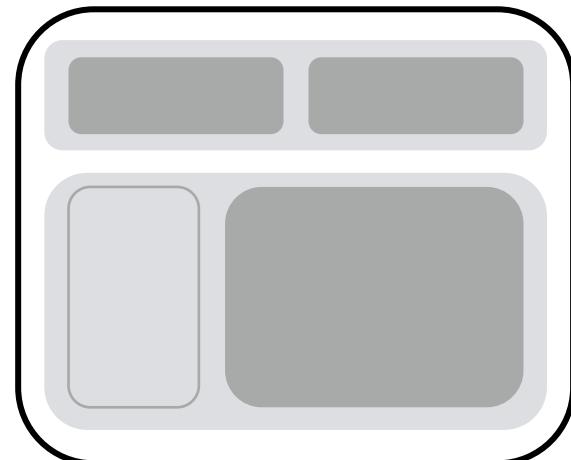
## Fixed Page

- Most limited
- Can only arrange elements in rows, which then get split into columns



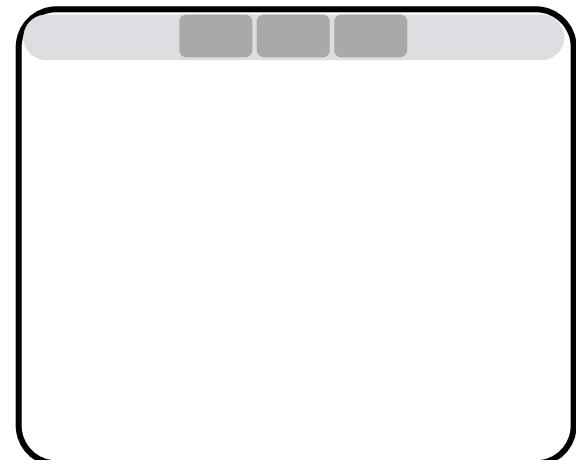
## Fluid Page

- Can arrange elements in rows, which then get split into columns
- Can also include other layouts and panels



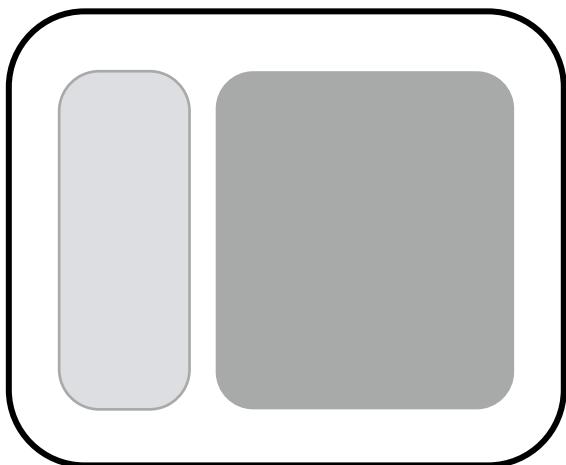
## Navbar Page

- Creates a navigation bar at the top
- Can specify a title
- Can add multiple pages, each of which is fluid



# Common Layout Options

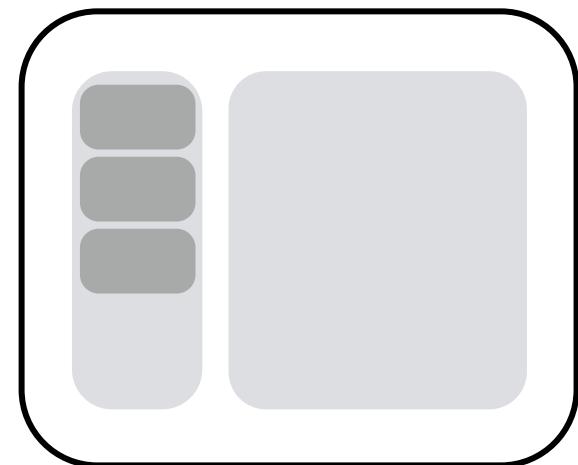
**Sidebar Layout**



**Tabset Panel**



**Navlist Panel**



# Additional Panels and Containers

- **Fixed or Fluid Row and Column**

For row-based pages, can organize content into rows, then split into columns. Column widths go from 1 to 12.

- **Title Panel**

Good way to provide a title to a page or tabset panel

- **Well Panel**

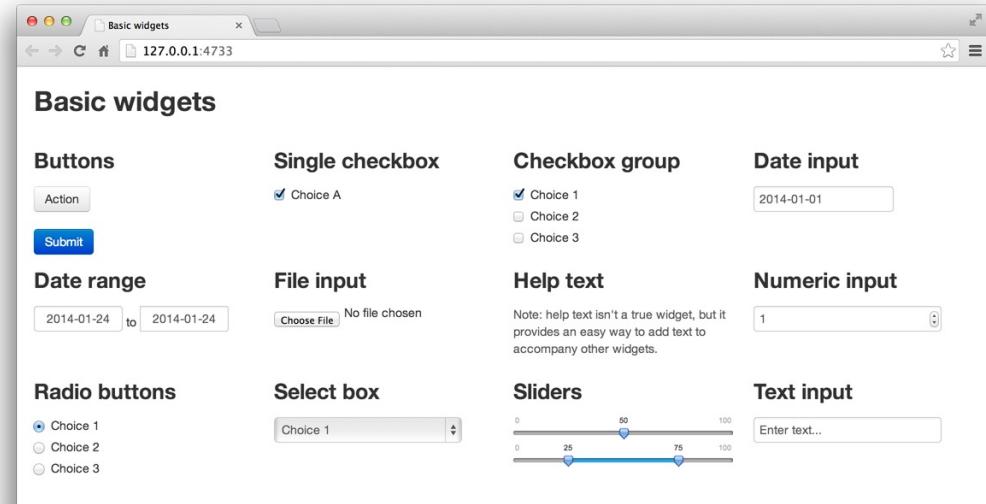
Basically a colored box you can put content in

- **Conditional Panel**

You can set this up so it is hidden until a condition is met

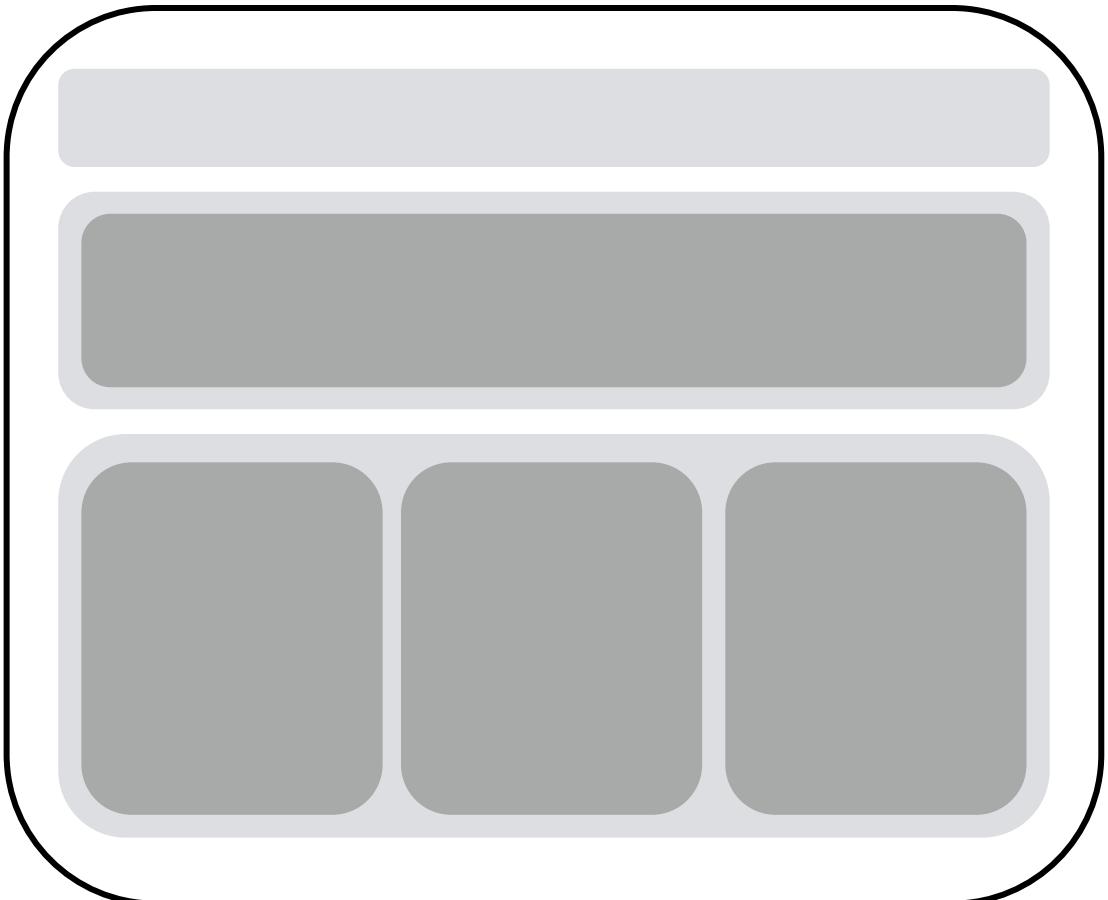
# What goes in the containers?

- Simple **static content**  
(like headers, text, images)
- Some kind of **input widget**  
(a way to get user feedback)
- Some kind of **render object**  
(a display created by R code)



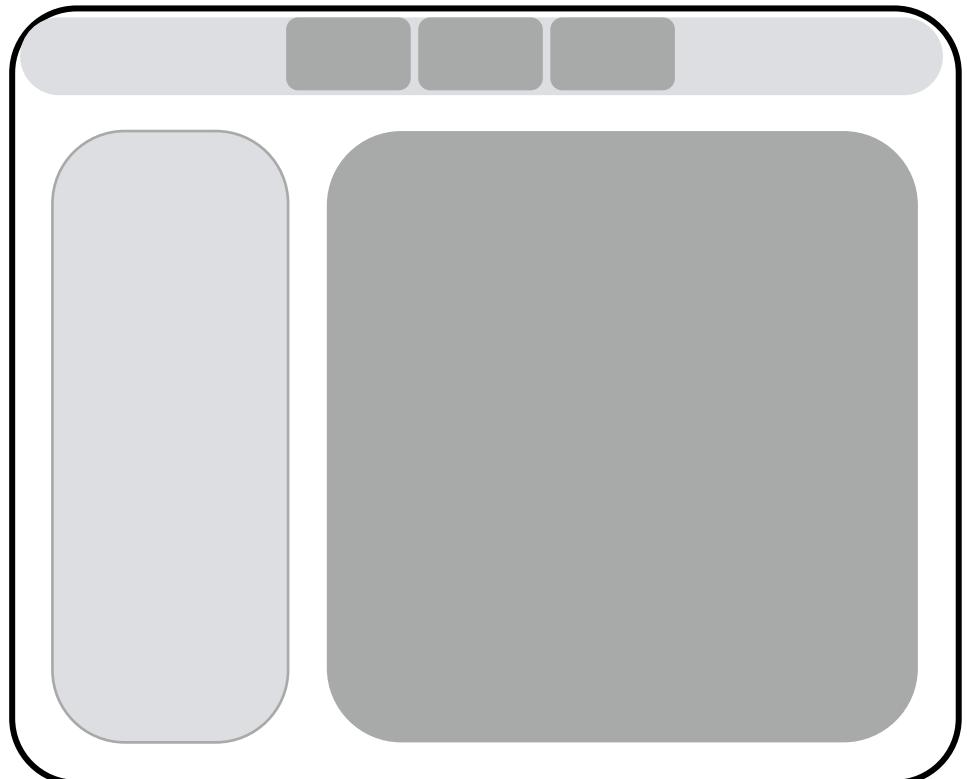
# Exercise 1: Basic Layout

- Create a fluid page
- Add a title panel
- Add two fluid rows, one with a single column, one with three columns
- Add static content (text and images) to the columns



# Exercise 2: Multi-Page Layout

- Create a navbar page
- Give it a title
- Add four tab panels
- Use each to test a different layout:
  - Sidebar Layout
  - Tabset Panel
  - Navlist Panel
  - Your choice
- Add static content to different parts of the layout to see where it shows up



# Different layout solution: Shiny Dashboard

[Background: Shiny and HTML](#)

[Structure overview](#)

**Header**

- [Message menus](#)
- [Dynamic content](#)
- [Notification menus](#)
- [Task menus](#)
- [Disabling the header](#)

**Sidebar**

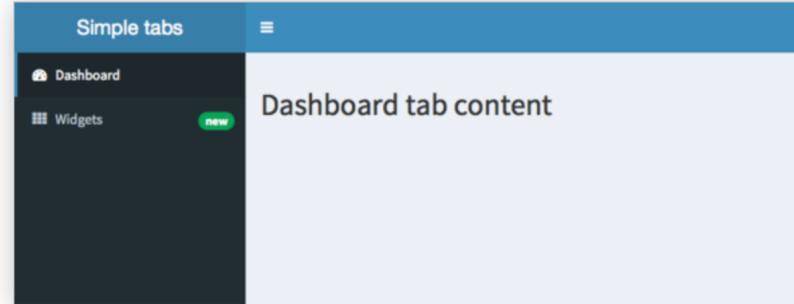
- [Sidebar menu items and tabs](#)
- [Bookmarking and restoring selected tabs](#)
- [Dynamic content](#)
- [Inputs in the sidebar](#)
- [Disabling the sidebar](#)

**Body**

- [Boxes](#)
- [tabBox](#)
- [infoBox](#)
- [valueBox](#)
- Layouts**
- [Row-based layout](#)
- [Column-based layout](#)
- [Mixed row and column layout](#)

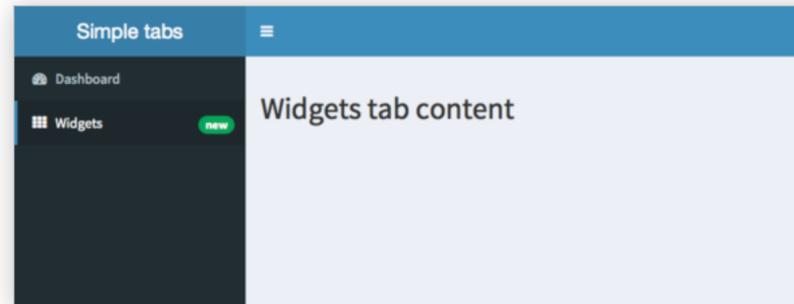
## Sidebar menu items and tabs

Links in the sidebar can be used like `tabPanels` from Shiny. That is, when you click on a link, it will display different content in the body of the dashboard. Here is an example of a simple `tabPanel`:



Tabs

When the user clicks on one of the menu items, it switches content in the main body:



<https://rstudio.github.io/shinydashboard/>

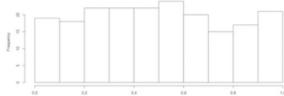
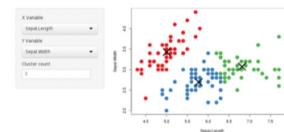
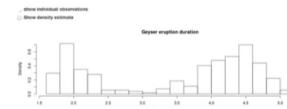
# Example apps

- [Shiny for R Gallery](#)
- [2019 Shiny Contest Winners](#)
- [2020 Shiny Contest Winners](#)
- [2021 Shiny Contest Winners](#)

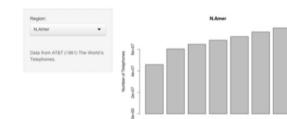


## Start Simple

If you're new to Shiny, these simple but complete applications are designed for you to learn from.



Faithful



Kmeans example



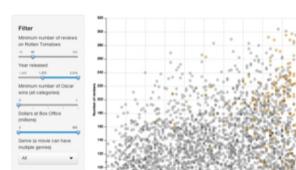
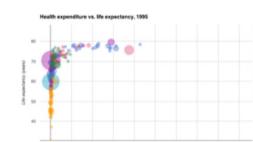
Single-file shiny app

Telephones by region

Word cloud

## Interactive visualizations

Shiny is designed for fully interactive visualization, using JavaScript libraries like [d3](#), [Leaflet](#), and [Google Charts](#).



# Guess the Layout: Pet Records

**Pet Records**

Select pet:

Layla  
 Lloyd



DOB: 07-21-2009  
Species: canine  
Breed: basset/boxer mix  
Sex: spayed female  
Color: white with brown spots

Weight History (lbs.)

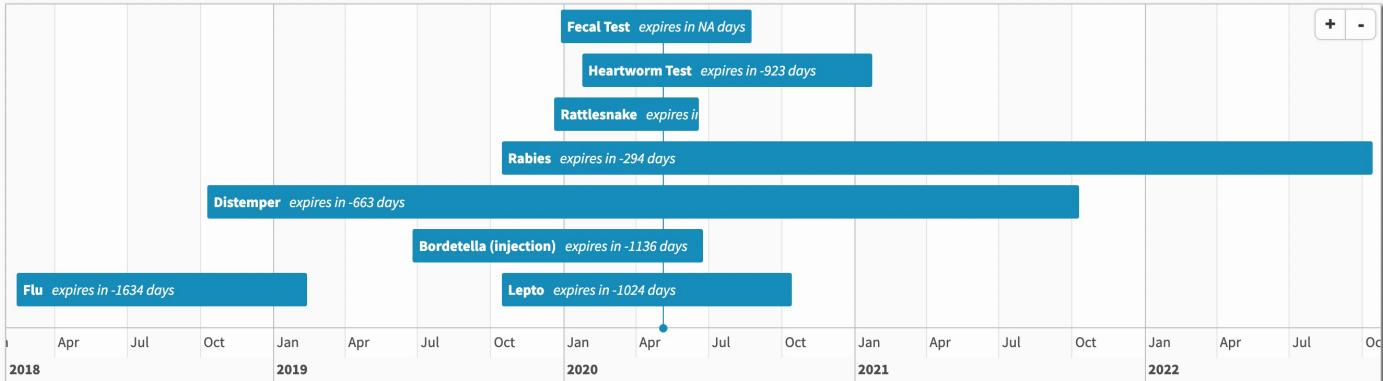


Date	Weight
12-20-2019	47.6
11-06-2019	49.0
10-15-2019	50.0
06-25-2019	51.2
02-09-2019	48.9
10-10-2018	47.2

Current Vaccines  
 Past Vaccines

**Vaccine Timeline**

Click an item to view vaccine certificate (where available). The information is shown below the timeline.



Fecal Test expires in NA days  
Heartworm Test expires in -923 days  
Rattlesnake expires in  
Rabies expires in -294 days  
Distemper expires in -663 days  
Bordetella (injection) expires in -1136 days  
Flu expires in -1634 days  
Lepto expires in -1024 days

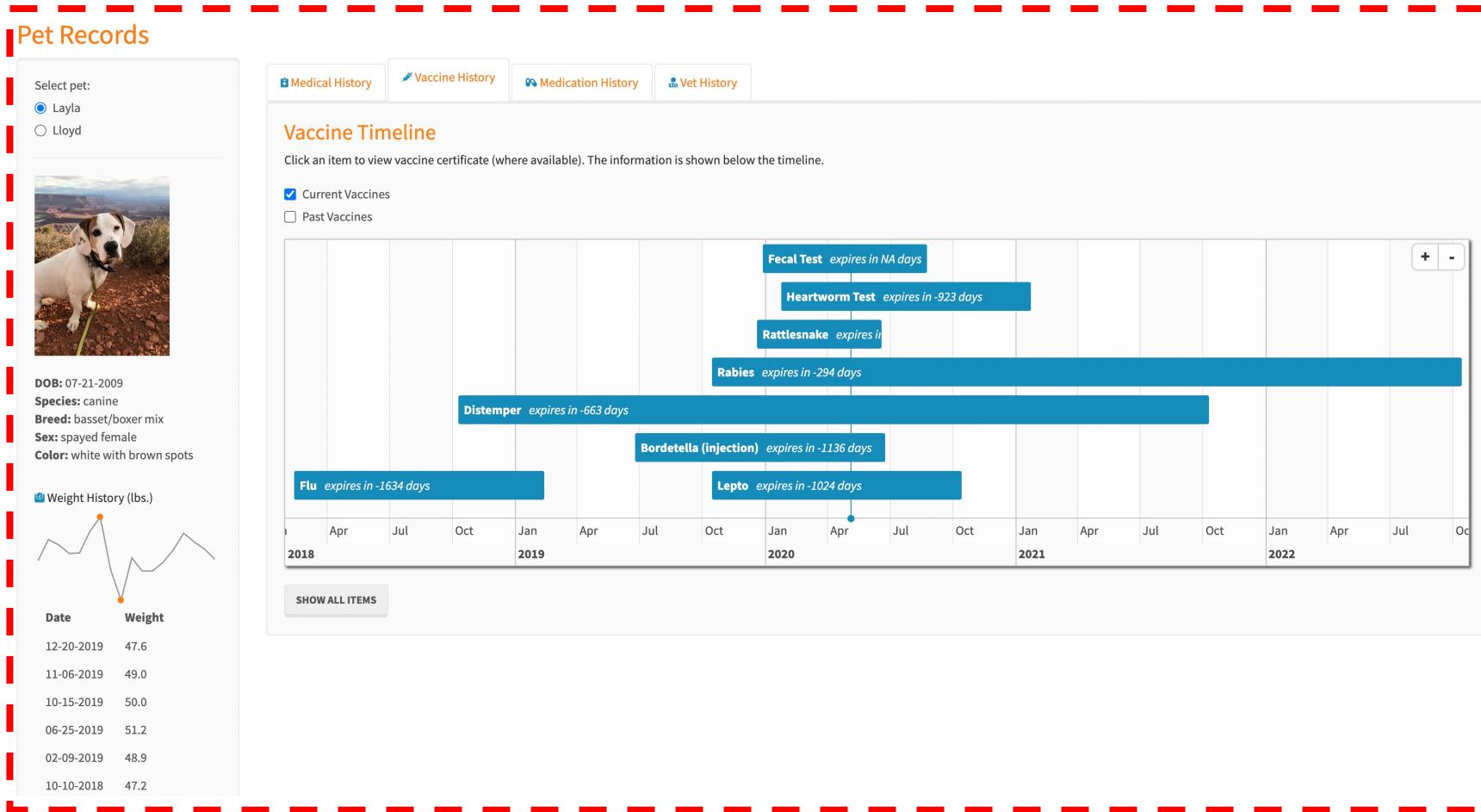
Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct 2018 2019 2020 2021 2022

**SHOW ALL ITEMS**

<https://jennadallen.shinyapps.io/pet-records-app/>

# Pet Records Answer

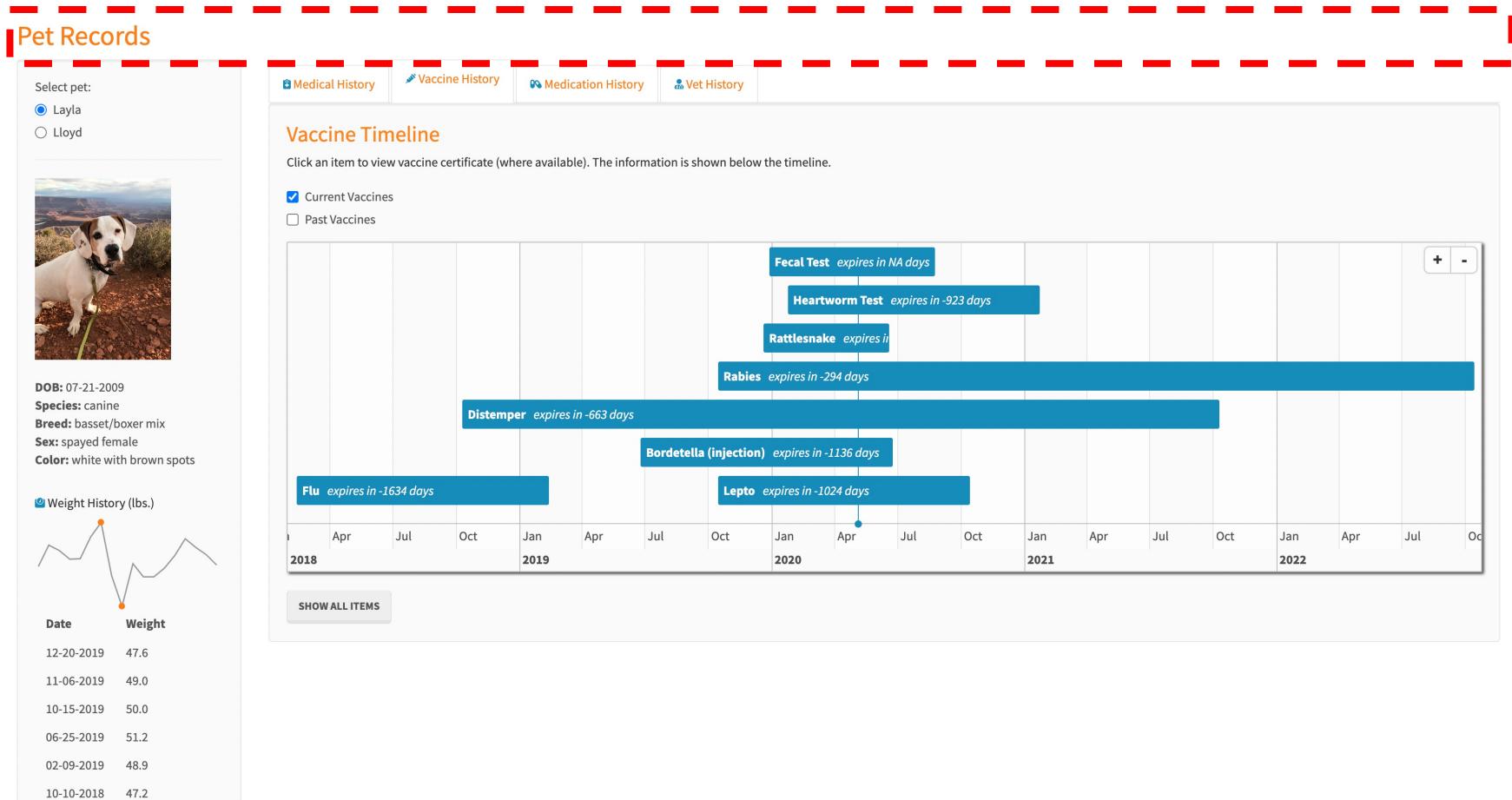
Fluid Page



[https://github.com/jennaallen/dog\\_days](https://github.com/jennaallen/dog_days)

# Pet Records Answer

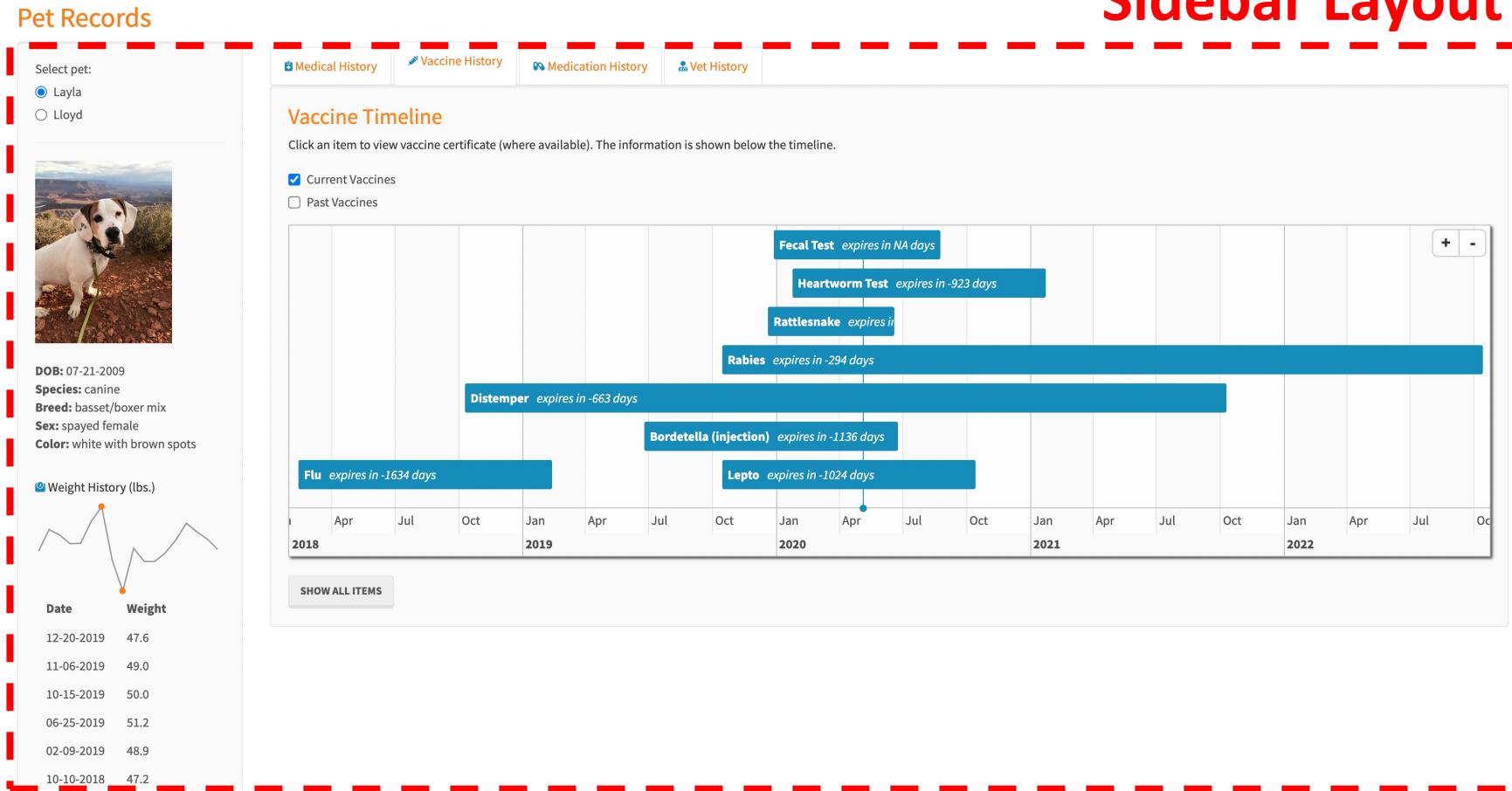
## Title Panel



[https://github.com/jennaallen/dog\\_days](https://github.com/jennaallen/dog_days)

# Pet Records Answer

## Sidebar Layout



[https://github.com/jennaallen/dog\\_days](https://github.com/jennaallen/dog_days)

# Pet Records Answer

Sidebar  
Panel

**Pet Records**

Select pet:

Layla  
 Lloyd



DOB: 07-21-2009  
Species: canine  
Breed: basset/boxer mix  
Sex: spayed female  
Color: white with brown spots

Weight History (lbs.)

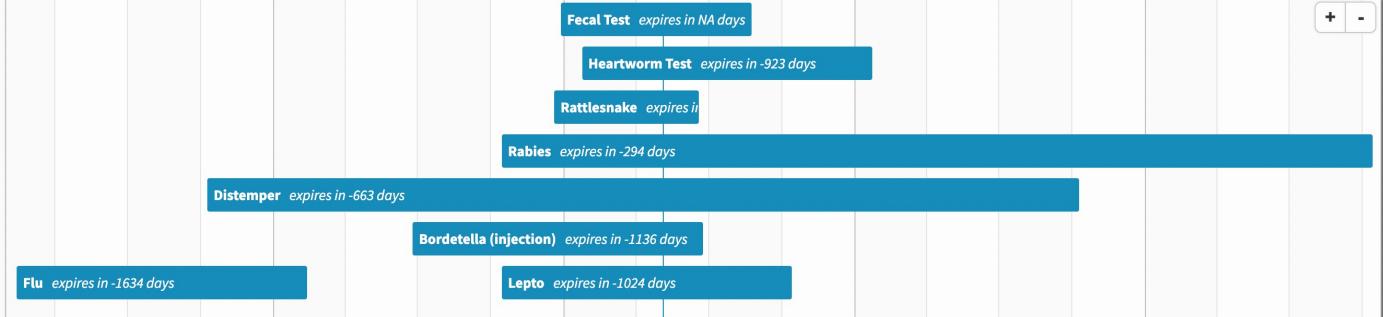


Date	Weight
12-20-2019	47.6
11-06-2019	49.0
10-15-2019	50.0
06-25-2019	51.2
02-09-2019	48.9
10-10-2018	47.2

**Vaccine Timeline**

Click an item to view vaccine certificate (where available). The information is shown below the timeline.

Current Vaccines  
 Past Vaccines



Rabies expires in -294 days  
Distemper expires in -663 days  
Flu expires in -1634 days  
Heartworm Test expires in -923 days  
Fecal Test expires in NA days  
Rattlesnake expires in  
Bordetella (injection) expires in -1136 days  
Lepto expires in -1024 days

Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct 2018 2019 2020 2021 2022

**SHOW ALL ITEMS**

[https://github.com/jennaallen/dog\\_days](https://github.com/jennaallen/dog_days)

# Pet Records Answer

**Pet Records**

Select pet:

Layla  
 Lloyd



DOB: 07-21-2009  
Species: canine  
Breed: basset/boxer mix  
Sex: spayed female  
Color: white with brown spots

Weight History (lbs.)

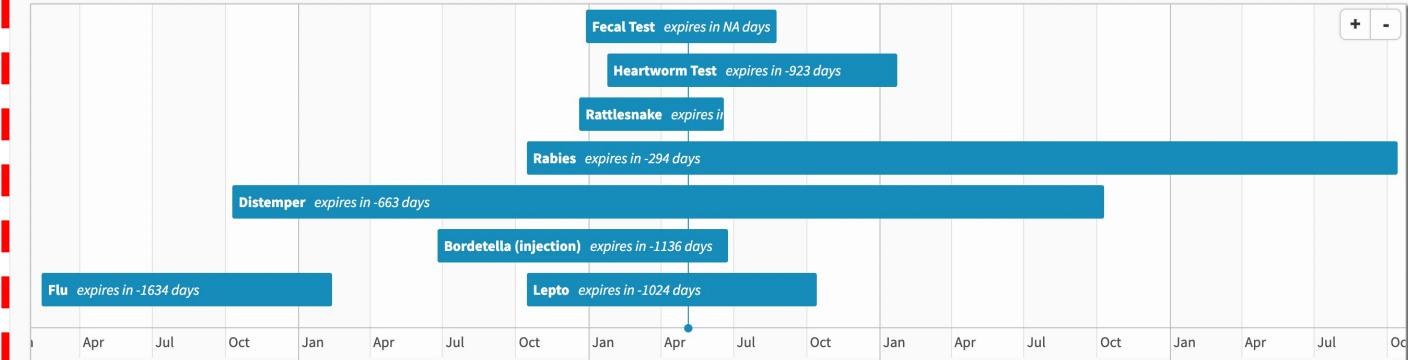


Date	Weight
12-20-2019	47.6
11-06-2019	49.0
10-15-2019	50.0
06-25-2019	51.2
02-09-2019	48.9
10-10-2018	47.2

**Vaccine Timeline**

Click an item to view vaccine certificate (where available). The information is shown below the timeline.

Current Vaccines  
 Past Vaccines



Fecal Test expires in NA days  
Heartworm Test expires in -923 days  
Rattlesnake expires in  
Rabies expires in -294 days  
Distemper expires in -663 days  
Bordetella (injection) expires in -1136 days  
Flu expires in -1634 days  
Lepto expires in -1024 days

Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct 2018 2019 2020 2021 2022

**Main Panel**

[https://github.com/jennaallen/dog\\_days](https://github.com/jennaallen/dog_days)

# Pet Records Answer

## Pet Records

Select pet:  
 Layla  
 Lloyd



DOB: 07-21-2009  
Species: canine  
Breed: basset/boxer mix  
Sex: spayed female  
Color: white with brown spots

Weight History (lbs.)

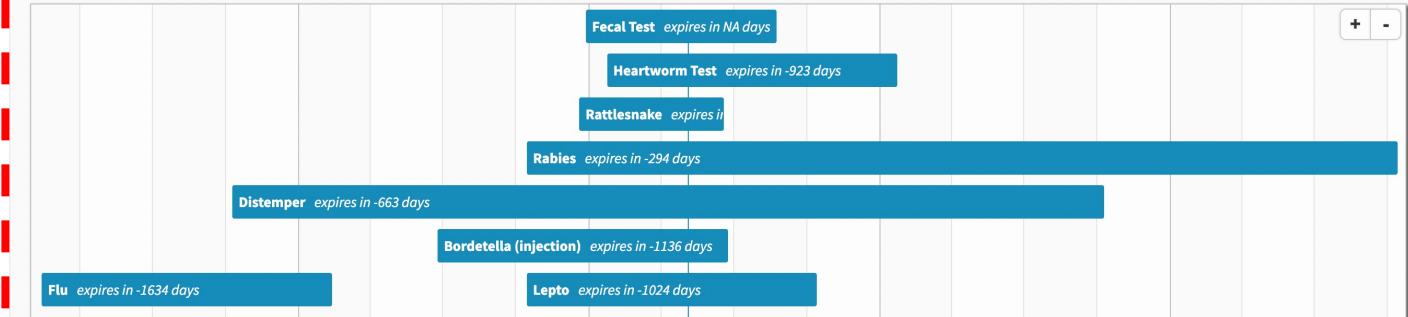


Date	Weight
12-20-2019	47.6
11-06-2019	49.0
10-15-2019	50.0
06-25-2019	51.2
02-09-2019	48.9
10-10-2018	47.2

## Vaccine Timeline

Click an item to view vaccine certificate (where available). The information is shown below the timeline.

Current Vaccines  
 Past Vaccines



Fecal Test expires in NA days  
Heartworm Test expires in -923 days  
Rattlesnake expires in -1136 days  
Rabies expires in -294 days  
Distemper expires in -663 days  
Bordetella (injection) expires in -1136 days  
Flu expires in -1634 days  
Lepto expires in -1024 days

Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct 2018 2019 2020 2021 2022

[SHOW ALL ITEMS](#)

## Tabset Panel

[https://github.com/jennaallen/dog\\_days](https://github.com/jennaallen/dog_days)

# Pet Records Answer

## Pet Records

Select pet:

Layla  
 Lloyd



DOB: 07-21-2009  
Species: canine  
Breed: basset/boxer mix  
Sex: spayed female  
Color: white with brown spots

Weight History (lbs.)

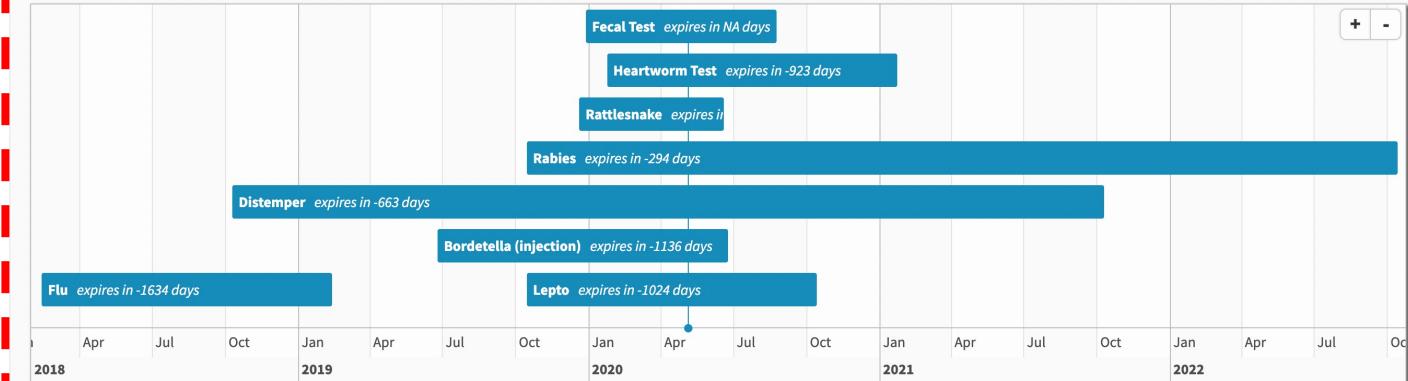


Date	Weight
12-20-2019	47.6
11-06-2019	49.0
10-15-2019	50.0
06-25-2019	51.2
02-09-2019	48.9
10-10-2018	47.2

## Vaccine Timeline

Click an item to view vaccine certificate (where available). The information is shown below the timeline.

Current Vaccines  
 Past Vaccines



Fecal Test expires in NA days  
Heartworm Test expires in -923 days  
Rattlesnake expires in  
Rabies expires in -294 days  
Distemper expires in -663 days  
Bordetella (injection) expires in -1136 days  
Flu expires in -1634 days  
Lepto expires in -1024 days

[SHOW ALL ITEMS](#)

## Tab Panel

[https://github.com/jennaallen/dog\\_days](https://github.com/jennaallen/dog_days)

# Pet Records Answer

## Pet Records

Select pet:

Layla  
 Lloyd



DOB: 07-21-2009  
Species: canine  
Breed: basset/boxer mix  
Sex: spayed female  
Color: white with brown spots

Weight History (lbs.)

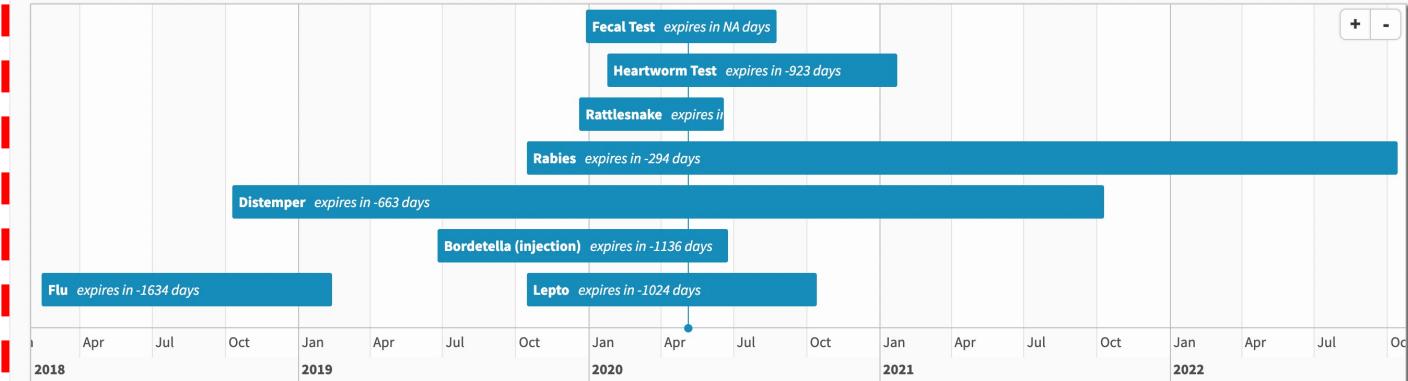


Date	Weight
12-20-2019	47.6
11-06-2019	49.0
10-15-2019	50.0
06-25-2019	51.2
02-09-2019	48.9
10-10-2018	47.2

## Vaccine Timeline

Click an item to view vaccine certificate (where available). The information is shown below the timeline.

Current Vaccines  
 Past Vaccines



Fecal Test expires in NA days  
Heartworm Test expires in -923 days  
Rattlesnake expires in  
Rabies expires in -294 days  
Distemper expires in -663 days  
Bordetella (injection) expires in -1136 days  
Flu expires in -1634 days  
Lepto expires in -1024 days

Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct 2018 2019 2020 2021 2022

[SHOW ALL ITEMS](#)

## Tab Panel

[https://github.com/jennaallen/dog\\_days](https://github.com/jennaallen/dog_days)

# Pet Records Answer

## Pet Records

Select pet:

Layla  
 Lloyd



DOB: 07-21-2009  
Species: canine  
Breed: basset/boxer mix  
Sex: spayed female  
Color: white with brown spots

Weight History (lbs.)

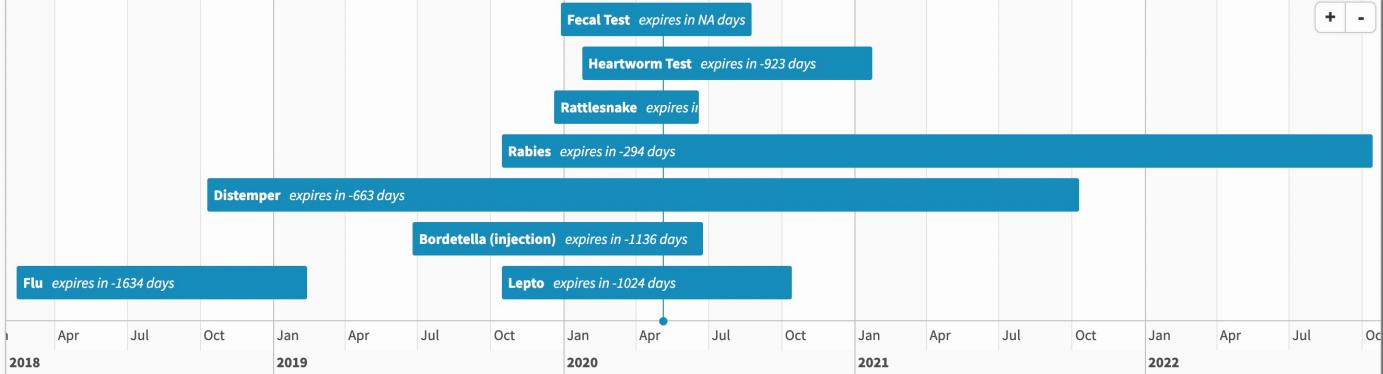


Date	Weight
12-20-2019	47.6
11-06-2019	49.0
10-15-2019	50.0
06-25-2019	51.2
02-09-2019	48.9
10-10-2018	47.2

Current Vaccines  
 Past Vaccines

### Vaccine Timeline

Click an item to view vaccine certificate (where available). The information is shown below the timeline.



Fecal Test expires in NA days  
Heartworm Test expires in -923 days  
Rattlesnake expires in  
Rabies expires in -294 days  
Distemper expires in -663 days  
Bordetella (injection) expires in -1136 days  
Flu expires in -1634 days  
Lepto expires in -1024 days

Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct 2018 2019 2020 2021 2022

[SHOW ALL ITEMS](#)

## Tab Panel

[https://github.com/jennaallen/dog\\_days](https://github.com/jennaallen/dog_days)

# Pet Records Answer

## Pet Records

Select pet:

Layla  
 Lloyd



DOB: 07-21-2009  
Species: canine  
Breed: basset/boxer mix  
Sex: spayed female  
Color: white with brown spots

Weight History (lbs.)

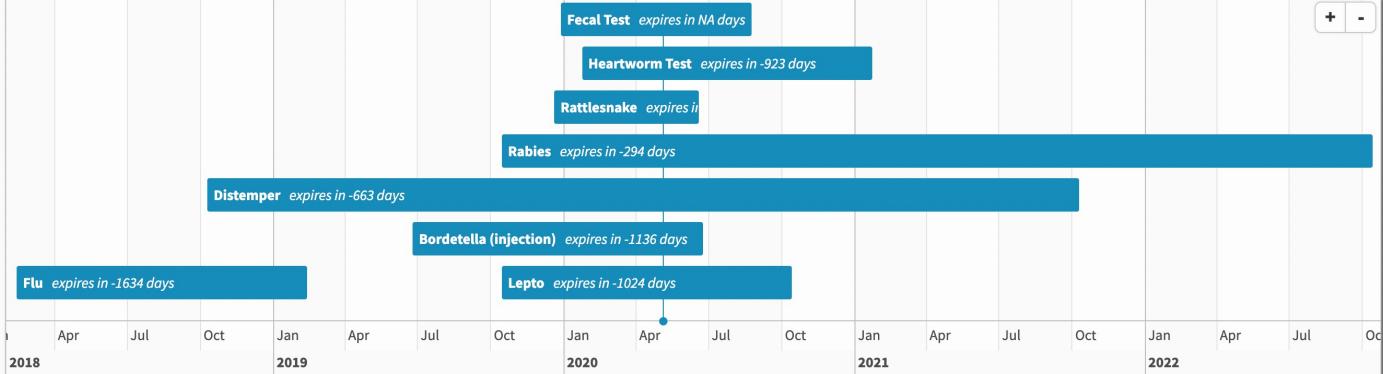


Date	Weight
12-20-2019	47.6
11-06-2019	49.0
10-15-2019	50.0
06-25-2019	51.2
02-09-2019	48.9
10-10-2018	47.2

### Vaccine Timeline

Click an item to view vaccine certificate (where available). The information is shown below the timeline.

Current Vaccines  
 Past Vaccines



Fecal Test expires in NA days  
Heartworm Test expires in -923 days  
Rattlesnake expires in  
Rabies expires in -294 days  
Distemper expires in -663 days  
Bordetella (injection) expires in -1136 days  
Flu expires in -1634 days  
Lepto expires in -1024 days

Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct 2018 2019 2020 2021 2022

[SHOW ALL ITEMS](#)

## Tab Panel

[https://github.com/jennaallen/dog\\_days](https://github.com/jennaallen/dog_days)

# Guess the Layout: 69 Love Songs



<https://committedtotape.shinyapps.io/sixtyninelovesongs/>

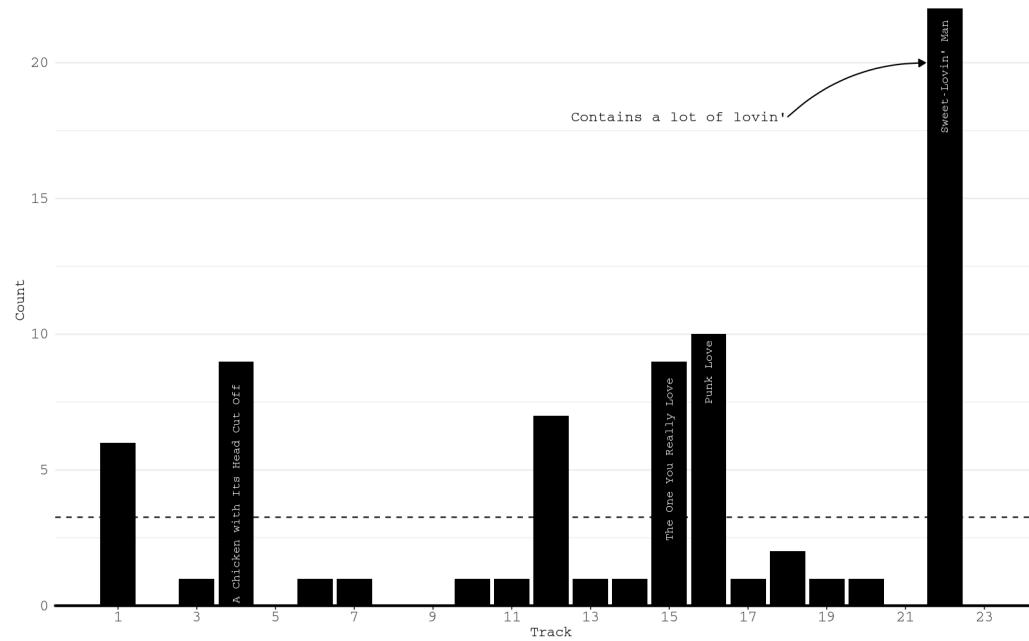
# 69 Love Songs Answer

Navbar Page

69 Love Songs    Intro    **How F%>%king Romantic?**    The Book of Love    Love, or Nothing At All    Love and Trouble    Making Me Blue

"How Fucking Romantic, must we really waltz?" 1.14 – How Fucking Romantic

But just how romantic is 69 Love Songs? Let's measure this (crudely) by the number of 'loves' in each song:



Select your Volume:

1

Info:

Volume 1 has an average love count of 3.26 per song.  
Indicated by the dotted line.

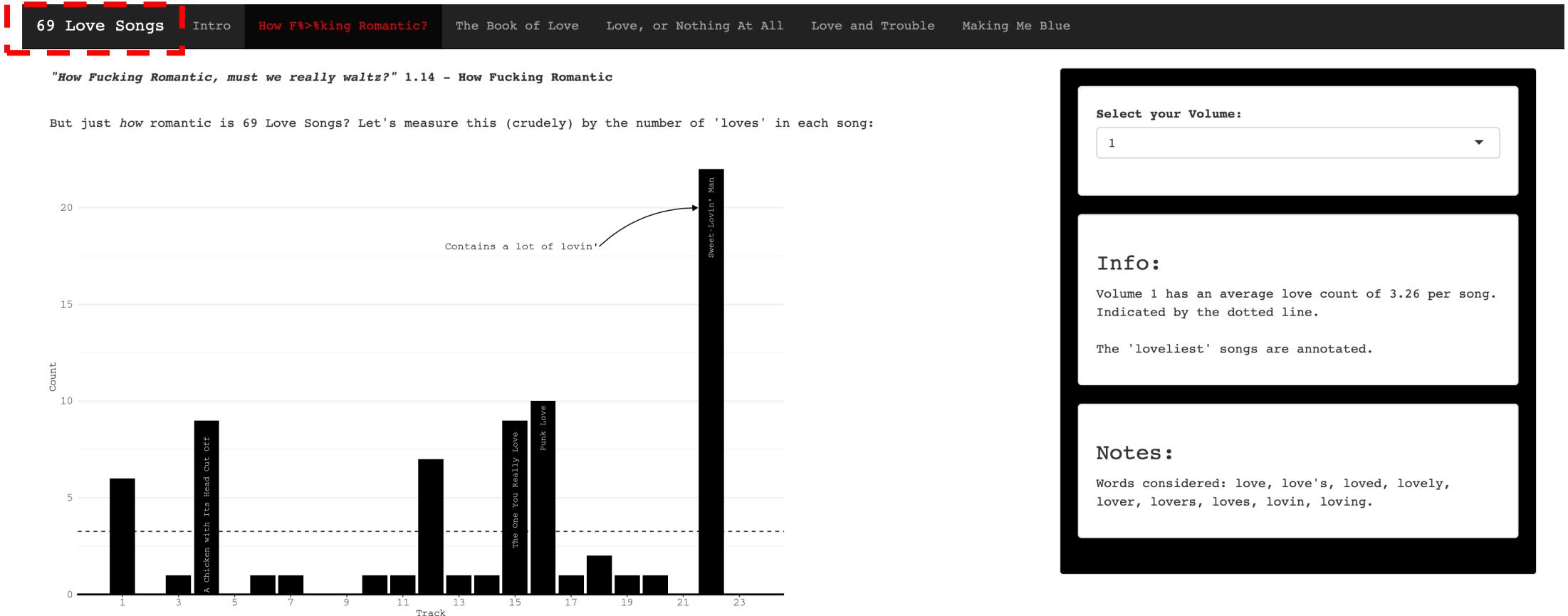
The 'loveliest' songs are annotated.

Notes:

Words considered: love, love's, loved, lovely,  
lover, lovers, loves, lovin, loving.

# 69 Love Songs Answer

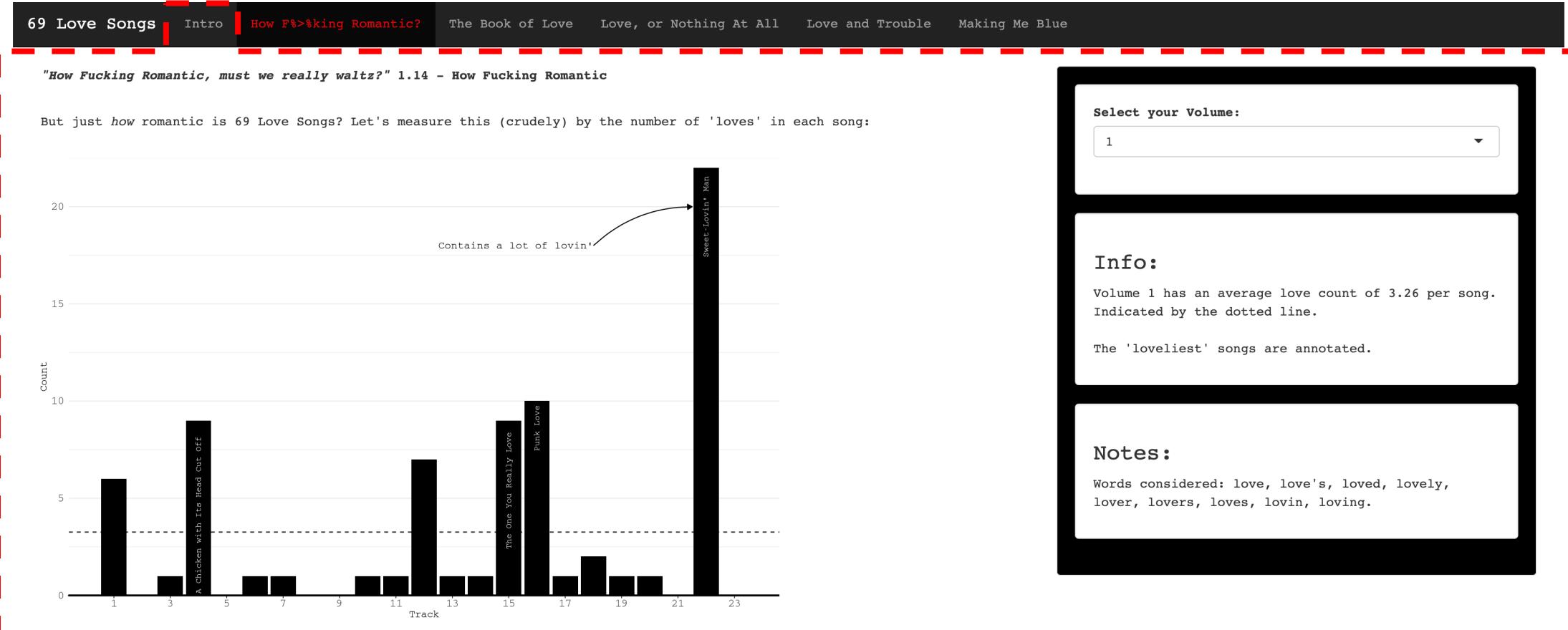
## Navbar Page Title



<https://committedtotape.shinyapps.io/sixtyninelovesongs/>

# 69 Love Songs Answer

## Tab Panel



<https://committedtotape.shinyapps.io/sixtyninelovesongs/>

# 69 Love Songs Answer

## Tab Panel



<https://committedtotape.shinyapps.io/sixtyninelovesongs/>

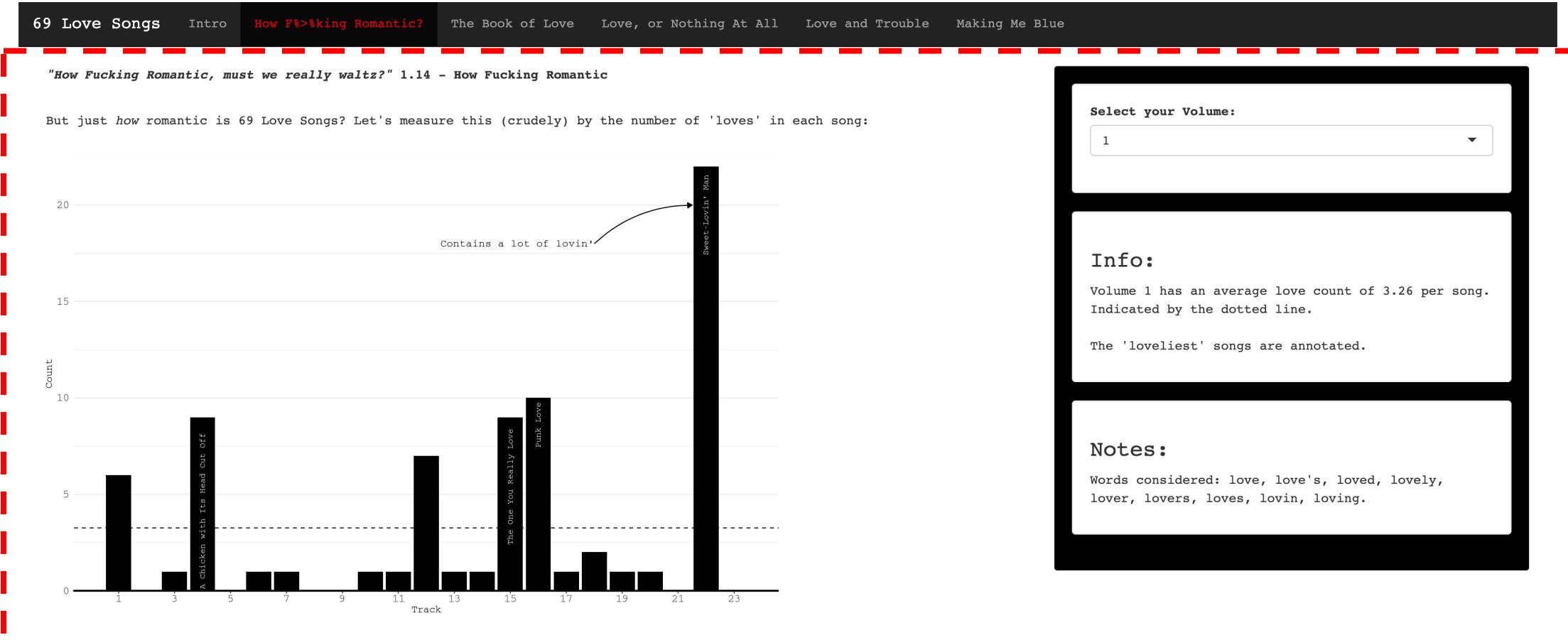
# 69 Love Songs Answer

Fluid Page



# 69 Love Songs Answer

## Sidebar Layout



# 69 Love Songs Answer

Sidebar Panel (position = "right")



<https://committedtotape.shinyapps.io/sixtyninelovesongs/>

# 69 Love Songs Answer

Well Panels



<https://committedtotape.shinyapps.io/sixtyninelovesongs/>

# 69 Love Songs Answer

## Main Panel



# 69 Love Songs Answer

## Tab Panels



<https://committedtotape.shinyapps.io/sixtyninelovesongs/>

Morning Break

Adding visual components or  
“outputs”

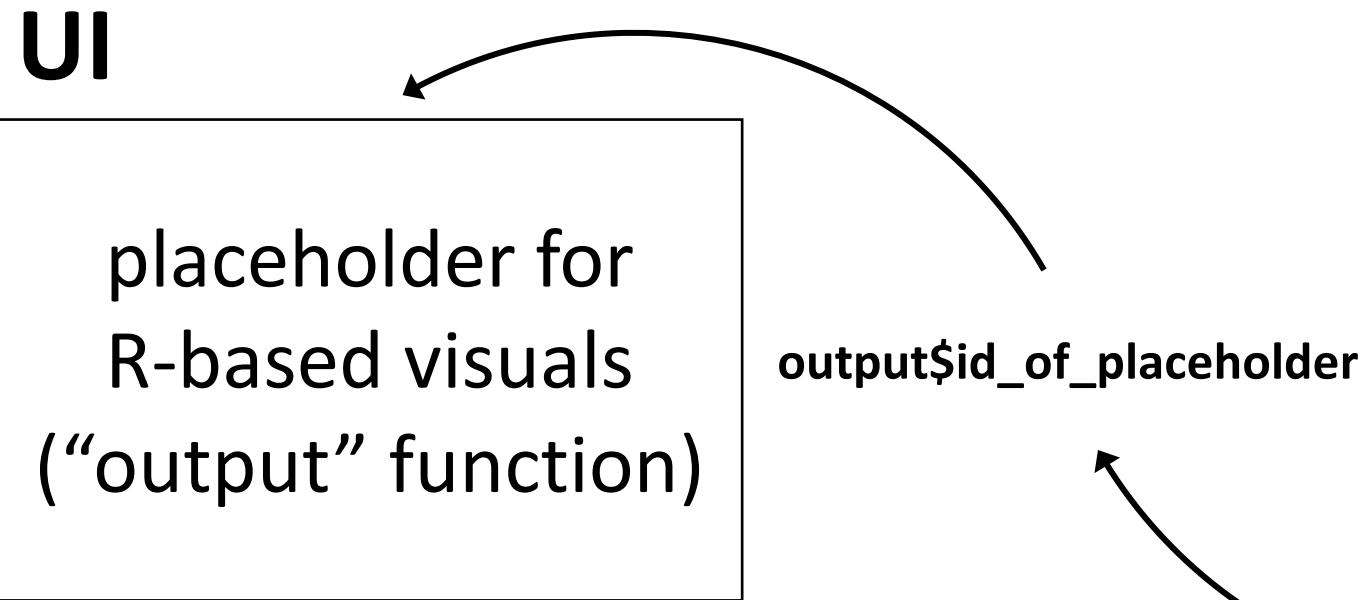
# How do we use R to create visual content?

So far, we have only been displaying static content, using functions inspired by HTML tags.

Our layout containers don't accept normal R code.  
(Why do you think this is?)

Where does the R code go? How does the result of the R code get displayed in the web app?

UI code specifies a placeholder to indicate where the R output should go and gives the output an identifier.



Server code saves R output to an object called “output”, using the same ID listed in the placeholder.

# Simple outputs and renders

Render objects with same names and types as the ones listed in output placeholders

**UI:**

```
textOutput(outputId = "text1")
```

**Server:**

```
output$text1 <- renderText({  
  "some text"  
})
```

# Output-Render Pairs

UI (output)	Server (render)
htmlOutput	renderUI
imageOutput	renderImage
plotOutput	renderPlot
tableOutput	renderTable
dataTableOutput	renderDataTable
textOutput	renderText
uiOutput	renderUI
verbatimTextOutput	renderPrint

<https://shiny.posit.co/r/getstarted/shiny-basics/lesson4/>

# Exercise 3: Display a Plot

- Create an app with a sidebar layout.
- Leave the sidebar empty for now.
- In the main panel, add a placeholder for a plot.
- In the server, add code for a plot.

# Exercise 4: Testing Different Outputs

- Create an app with a navigation bar.
- Create five pages to test different outputs:
  - Image output
  - Table output
  - Verbatim text output
  - Text output
  - HTML output

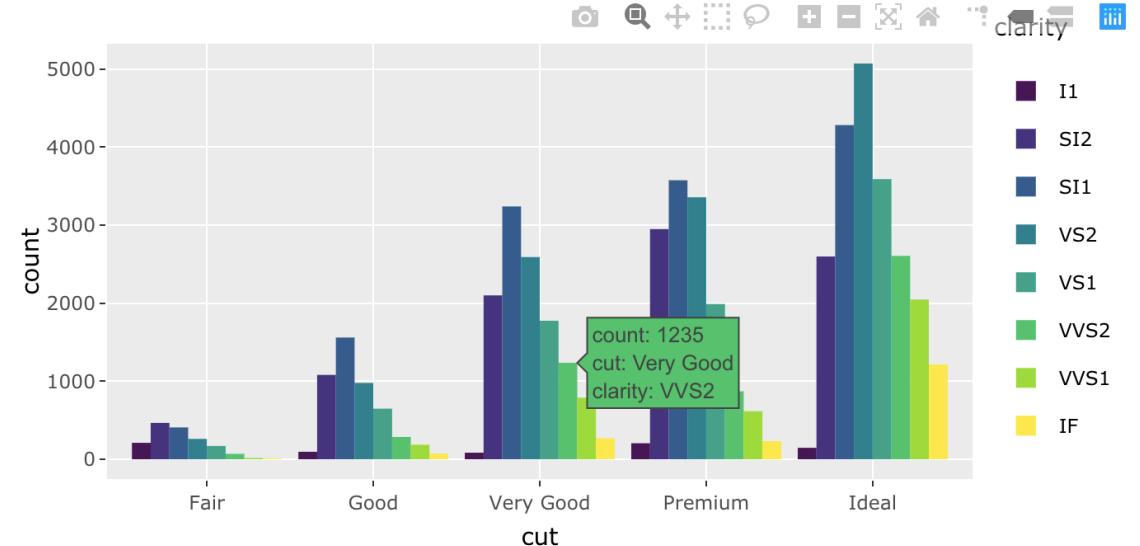
# Interactive components

See “InteractiveOutputs” in examples folder

# HTML widgets

- Some R packages can create interactive elements by outputting HTML/JavaScript code instead of normal R visuals
- These HTML widgets are useful for both R Markdown documents and Shiny apps, since both render HTML

<http://www.htmlwidgets.org/>



iris						
	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species	
1	5.1	3.5	1.4	0.2	setosa	
2	4.9	3.0	1.4	0.2	setosa	
3	4.7	3.2	1.3	0.2	setosa	
4	4.6	3.1	1.5	0.2	setosa	
5	5.0	3.6	1.4	0.2	setosa	

Show 5 entries Search:

Sepal.Length Sepal.Width Petal.Length Petal.Width Species

1 5.1 3.5 1.4 0.2 setosa

2 4.9 3.0 1.4 0.2 setosa

3 4.7 3.2 1.3 0.2 setosa

4 4.6 3.1 1.5 0.2 setosa

5 5.0 3.6 1.4 0.2 setosa

Showing 1 to 5 of 150 entries

Previous 1 2 3 4 5 ... 30 Next

Connecting inputs to outputs

# How do we allow user feedback to change visual elements in the app?

Now we have a way to inject the output of R code into our web app.

But if we are trying to help our user explore the data, we need to give them a way to change the display.

How can we design the app to accept user feedback? How can our R code respond to that feedback?

# Input widgets

- Button
- Checkboxes
- Date, date range input
- File input
- Numeric input
- Radio buttons
- Drop-down (select) box
- Slider, slider range
- Text input

The image shows a grid of six examples of shiny input widgets:

- Action button**: A button labeled "Action". Below it, a "Current Value:" box contains code: [1] 0 attr(,"class") [1] "shinyActionButtonValue" "integer". A "See Code" button is at the bottom.
- Single checkbox**: A checked checkbox labeled "Choice A". Below it, a "Current Value:" box contains code: [1] TRUE. A "See Code" button is at the bottom.
- Checkbox group**: Three checkboxes labeled "Choice 1", "Choice 2", and "Choice 3", with "Choice 1" checked. Below it, a "Current Values:" box contains code: [1] "1". A "See Code" button is at the bottom.
- Date input**: A text input field containing the value 2014-01-01. Below it, a "Current Value:" box contains code: [1] "2014-01-01". A "See Code" button is at the bottom.
- Date range**: Two text input fields for "from" and "to" dates, both containing 2023-08-04. Below it, a "Current Values:" box contains code: [1] "2023-08-04" "2023-08-04". A "See Code" button is at the bottom.
- File input**: A file selection input field showing "No file selected". Below it, a "Current Value:" box contains code: NULL. A "See Code" button is at the bottom.

<https://shiny.posit.co/r/getstarted/shiny-basics/lesson3/>  
<https://shiny.posit.co/r/gallery/widgets/widget-gallery/>

# Anatomy of an input widget

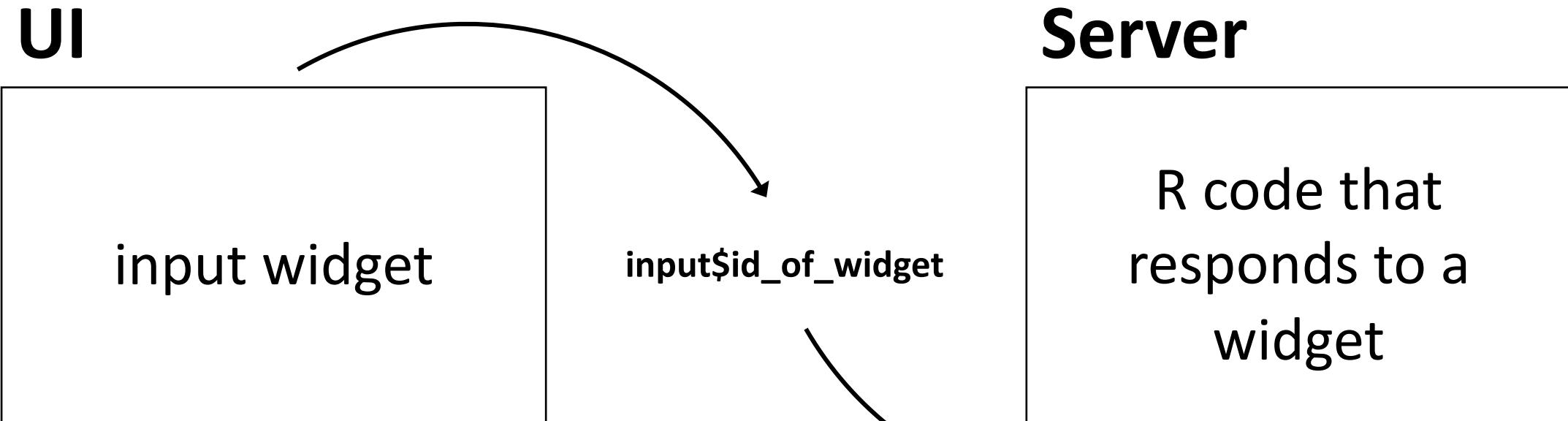
- **inputId** for the widget (internal only)
- **label** (will be visible)
- Check documentation for other required arguments (e.g., `selectInput` requires `choices`)

Select box



```
selectInput(inputId = "selectBox",
            label = "Select box:",
            choices = c("Choice 1",
                        "Choice 2",
                        "Choice 3"))
```

UI code configures the input widget  
and gives it an identifier.



Server code watches the object called  
“input”, using the same ID listed in the UI.  
R code is rerun when the input changes.

# Simple structure of inputs, outputs, and renders

- Render objects with same names and types as the ones listed in output placeholders
- Calls to input objects, using the same names as the control widgets

## UI:

```
sliderInput(inputId = "slider1",  
            ...)  
  
textOutput(outputId = "text1")
```

## Server:

```
output$text1 <- renderText({  
  input$slider1  
})
```

# Exercise 5: Basic Input

- Create an app with a sidebar layout.
- In the sidebar, add a drop-down menu with options 1, 2, and 3.
- In the main panel, add a placeholder for an image.
- In the server, use R code to render a different image depending on which option is selected in the drop-down.

Lunch break

Updating R code based on user  
feedback: Reactivity

# Reactivity

Writing R code for a Shiny app is a bit different from normal Shiny code.

We've already seen that Shiny has access to special objects called input and output.

Using these objects in the special way defined by Shiny allows the server code to respond or “react” to changes in inputs.

This is reactive programming.

<https://mastering-shiny.org/basic-reactivity.html>

# What have we learned so far?

## **User Interface (UI)**

the website people will see and interact with, including static content, inputs and (placeholders for) outputs

## **Server**

takes values from the inputs, does some calculations, and fills in the outputs

UI code specifies a placeholder to indicate where the R output should go and give the output an identifier.

**UI**

placeholder for  
R-based visuals  
("output" function)

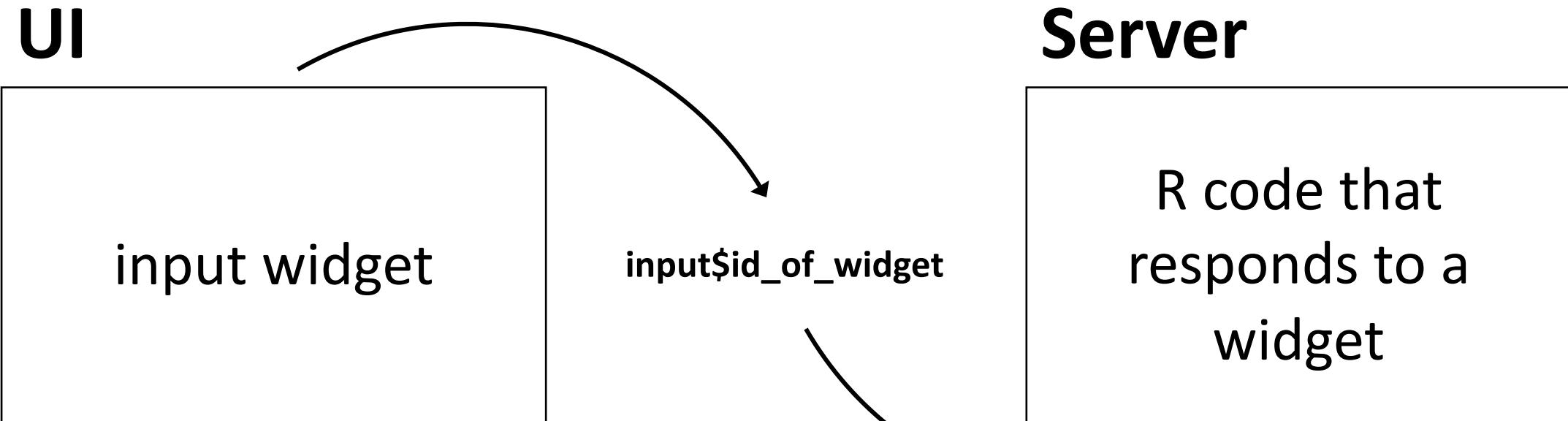
**Server**

R code that  
generates a visual  
("render" function)

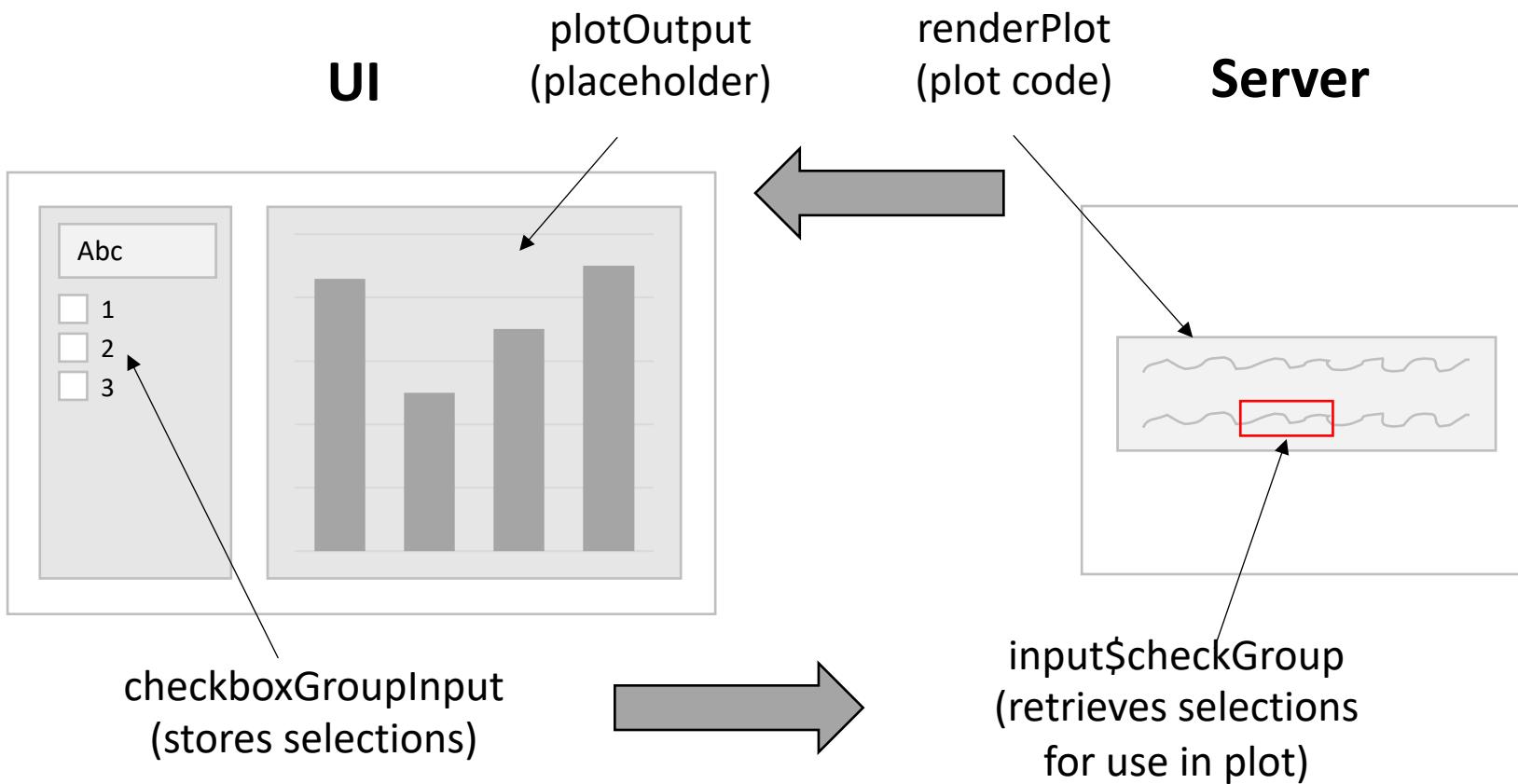
`output$id_of_placeholder`

Server code saves R output to an object called "output", using the same ID listed in the placeholder.

UI code configures the input widget  
and gives it an identifier.



Server code watches the object called  
“input”, using the same ID listed in the UI.  
R code is rerun when the input changes.



# Exercise 6: App from scratch

- In RStudio, use the File menu to create a new file
- Select “Shiny Web App...”
- Use any name you like
- Leave “Single file (app.R)” selected
- Select any directory you like
- Click “Create”
- Explore the app structure

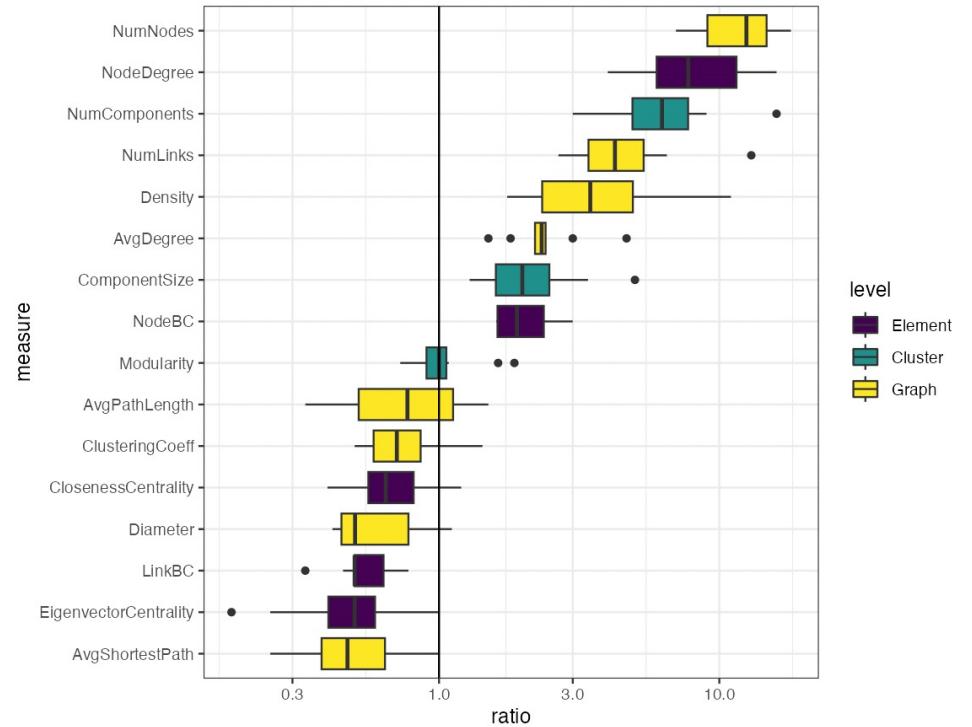
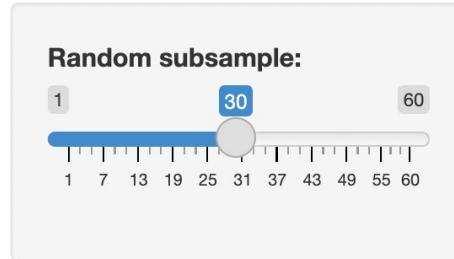
Discussion: What would you like  
to do with Shiny?

# Other examples (in “examples” folder)

May require additional packages

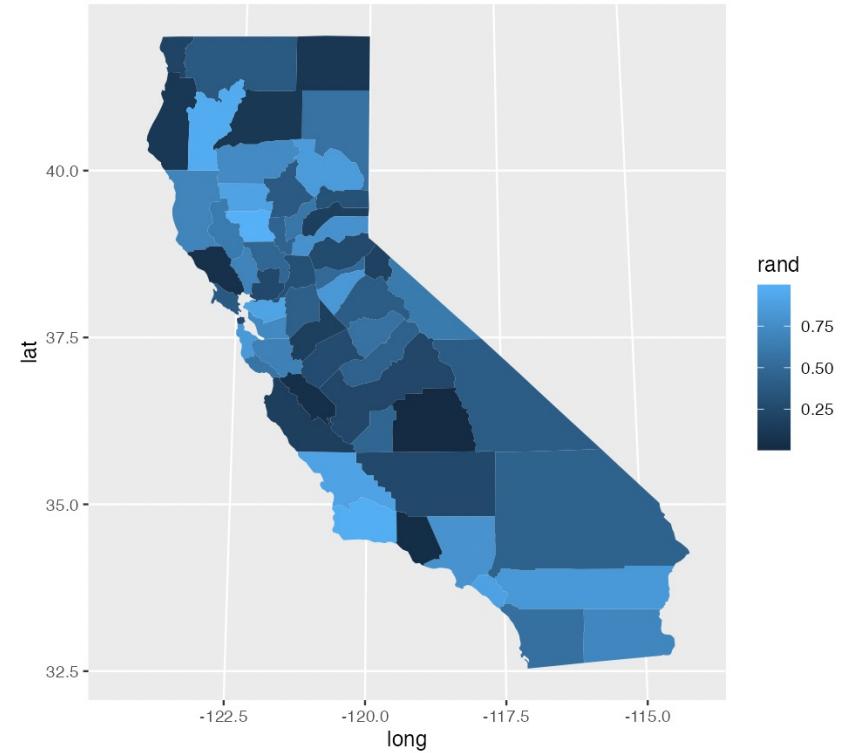
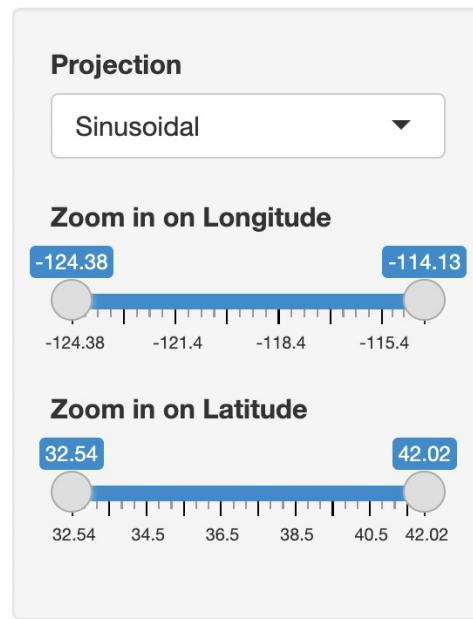
# Ratings

## Bootstrapping



# Mapping

## Projections Exercise



# Graph Builder

## ggplot Chart Builder

**Chart design**

Choose a variable for the X-axis:

Choose a variable for the Y-axis:

Choose a built-in theme:

**Annotation**

To place a text annotation on the graph, type the annotation below and click inside the chart to indicate the correct location.

Type an annotation:

Change the size of the text:

1

3

5

7

9

11

13

15

17

19

20

1

3

5

7

9

11

13

15

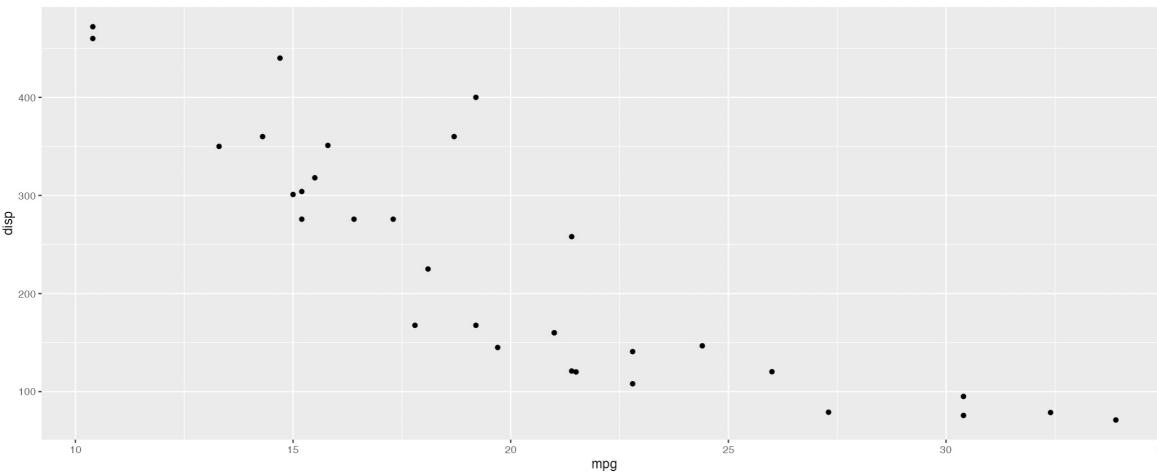
17

19

20

**Interactivity**

Make it interactive?



Code to create this plot:

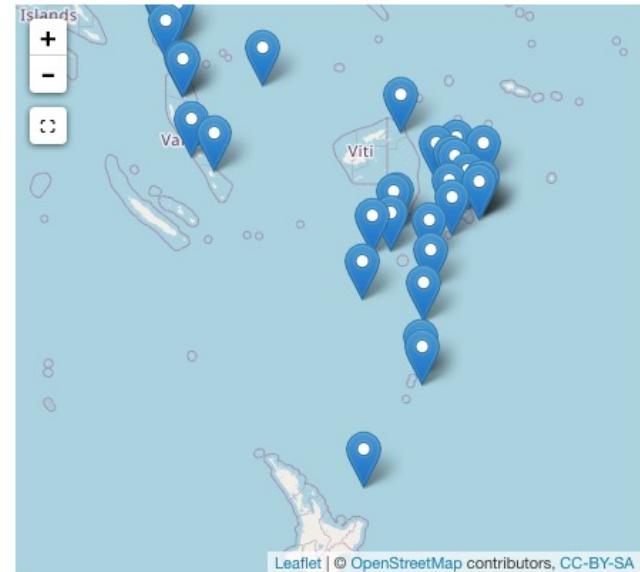
```
ggplot(mtcars, aes(x=mpg, y=disp)) + geom_point() + theme_grey()
```

Afternoon break

# Coordinated Views

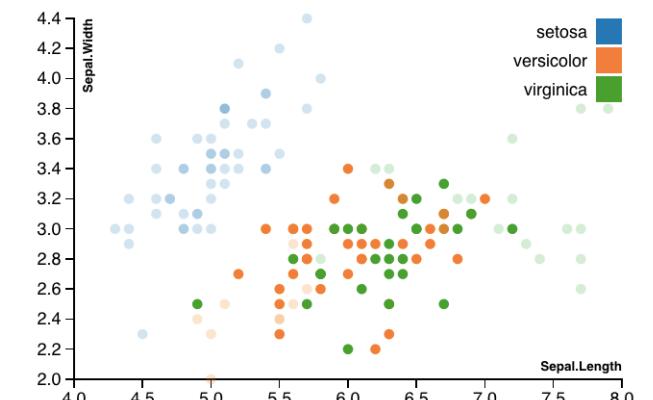
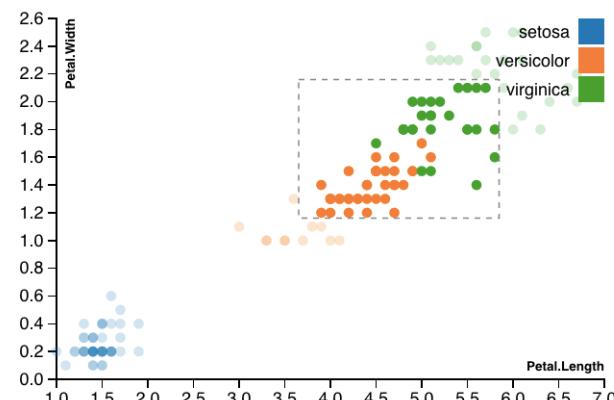
# Views that share data

- Each view should be relatively simple, have a specific purpose
- Views can work together to explore complex interactions
- How can Shiny help us display multiple views of the same data?



	lat↑	long↑	depth↓	mag↑	stations↑
308	-22	180.53	583	4.9	20
873	-11.02	167.01	62	4.9	36
277	-23.33	180.18	528	5	59
752	-21.29	185.77	57	5.3	69
352	-12.01	166.29	59	4.9	27
354	-30.17	182.02	56	5.5	68
168	-19.89	183.84	244	5.3	73
474	-10.79	166.06	142	5	40
338	-27.19	182.18	69	5.4	68

Showing 1 to 10 of 32 entries (filtered from 100 total entries)



# Option 1: Duplication

For each output that needs to be reactive to the input, repeat the same calculation.

```
output$a <- renderPlot({  
  ...  
  data %>%  
    filter(column == input$value)  
  ...  
})  
  
output$b <- renderTable({  
  ...  
  data %>%  
    filter(column == input$value)  
  ...  
})
```

# Option 2: reactive()

- Unbeknownst to us, we've been working in a “reactive context” every time we use a “render” function
- If we want to do some calculations in R that involve reactivity, but we don't necessarily want to create a visual output right away, we can use reactive() to store something that reacts to inputs
- **When you use something created by reactive(), you need to add () to the name**

```
variable <- reactive({  
  ...  
  data %>%  
    filter(column == input$value)  
  ...  
})  
  
output$a <- renderPlot({  
  ...  
  variable()  
  ...  
})  
  
output$b <- renderTable({  
  ...  
  variable()  
  ...  
})
```

# Exercise 7: Reactive Data

- Create an app with a sidebar layout
- Add a select input with all of the values of the mtcars “cyl” column
- Add a plot placeholder and a table (or data table) placeholder
- In the server, create a function that filters the mtcars dataset based on the value selected in the input
- Use this same dataset to render both the plot and table outputs

# Using charts as inputs

# Turning a plotOutput into an input

The plotOutput function has a few additional (powerful!) options.

- **click**: when someone clicks on the plot, store the x & y coordinates in input under the id specified
- **dblclick**: when someone double clicks, store the x & y coordinates
- **hover**: when someone hovers, store the x & y coordinates
- **brush**: when someone clicks and drags to draw a box on the plot, store the x and y coordinate ranges

# Additional related functions

When you add a click or brush to a plotOutput, you can use the following functions in the server code:

- **nearPoints()**: returns rows from a data frame which are near a click, hover, or double-click
- **brushedPoints()**: returns rows from a data frame which are under a brush

# Exercise 8: Plots and brushes

- Create an app with two plots
- Add a brush to one plot
- In the other plot, tie the brush to a layer that adds red points

How to learn more

# Advanced features and concepts

- [`reactiveVal\(\)`](#) and [`reactiveValues\(\)`](#): variables that are created in the server but can trigger code to re-run when updated
- [`observe\(\)`](#) and [`observeEvent\(\)`](#): server code that reacts to changes in inputs but doesn't get saved as a variable
- [`modules`](#): for complicated applications, you might want to break up your code into smaller chunks called modules
- [`debugging`](#): when you have trouble figuring out why your app isn't working, you might want to use some debugging options like `browser()` and `breakpoints`

# Shiny resources

- [Mastering Shiny book](#)
- [Shiny Tutorial](#)
- [Shiny Articles](#)
- [Shiny function reference](#)
- [Shinyapps.io](#)
- rstudio::conf 2022 workshop: [Getting Started with Shiny](#)
- [Shiny in Production \(slides\)](#), [Shiny in Production \(book\)](#)
- [Big Book of R: Shiny](#)

Thanks for your time today!

[angela.zoss@duke.edu](mailto:angela.zoss@duke.edu)