

SHINY WEB APPS

Heike Hofmann

THE



- Create conceptual understanding of shiny apps
- LOTS of practice
- Come out of the course with a shiny app of your own

SETUP

- Materials (pdf, code) are located in GitHub: XXX
Clone the directory onto your local machine
- Open an RStudio session (on Rivanna, locally or
RStudio cloud: XXX)

OUTLINE

- Day 1
 - Getting ready
 - Inputs & Outputs
 - Elements of Reactivity
 - Layouts
- Day 2
 - leaflets in shiny
 - Work on shiny apps for the DSPG projects

CHECKING VERSIONS OF PACKAGES

- Make sure you have package devtools installed:
`library(devtools)`
- If the command fails, install the package
`install.packages("devtools")`

SESSION INFO

```
library(devtools)
session_info()
```

*Whenever you have a
strange problem, make a
copy of your session info*

```
> session_info()
- Session info -
  setting  value
  version   R version 4.0.1 (2020-06-06)
  os        macOS Catalina 10.15.5
  system    x86_64, darwin17.0
  ui        RStudio
  language (EN)
  collate   en_US.UTF-8
  ctype     en_US.UTF-8
  tz        America/Chicago
  date      2020-06-27
```

Packages					
package	*	version	date	lib	source
assertthat	0.2.1	2019-03-21	[2]	CRAN (R 4.0.0)	
backports	1.1.7	2020-05-13	[2]	CRAN (R 4.0.0)	
callr	3.4.3	2020-03-28	[2]	CRAN (R 4.0.0)	
cli	2.0.2	2020-02-28	[2]	CRAN (R 4.0.0)	
crayon	1.3.4	2017-09-16	[2]	CRAN (R 4.0.0)	
desc	1.2.0	2018-05-01	[2]	CRAN (R 4.0.0)	
devtools	* 2.3.0	2020-04-10	[1]	CRAN (R 4.0.0)	
digest	0.6.25	2020-02-23	[2]	CRAN (R 4.0.0)	
ellipsis	0.3.1	2020-05-15	[2]	CRAN (R 4.0.0)	
fansi	0.4.1	2020-01-08	[2]	CRAN (R 4.0.0)	
fs	1.4.1	2020-04-04	[1]	CRAN (R 4.0.0)	
glue	1.4.1	2020-05-13	[2]	CRAN (R 4.0.0)	
magrittr	1.5	2014-11-22	[2]	CRAN (R 4.0.0)	
memoise	1.1.0	2017-04-21	[2]	CRAN (R 4.0.0)	
pkgbuild	1.0.8	2020-05-07	[2]	CRAN (R 4.0.0)	
pkgload	1.1.0	2020-05-29	[2]	CRAN (R 4.0.0)	
prettyunits	1.1.1	2020-01-24	[2]	CRAN (R 4.0.0)	
processx	3.4.2	2020-02-09	[2]	CRAN (R 4.0.0)	
ps	1.3.3	2020-05-08	[2]	CRAN (R 4.0.0)	
R6	2.4.1	2019-11-12	[2]	CRAN (R 4.0.0)	
remotes	2.1.1	2020-02-15	[1]	CRAN (R 4.0.0)	
rlang	0.4.6	2020-05-02	[2]	CRAN (R 4.0.0)	
rprojroot	1.3-2	2018-01-03	[2]	CRAN (R 4.0.0)	
rstudioapi	0.11	2020-02-07	[2]	CRAN (R 4.0.0)	
sessioninfo	1.1.1	2018-11-05	[1]	CRAN (R 4.0.0)	
testthat	2.3.2	2020-03-02	[2]	CRAN (R 4.0.0)	
usethis	* 1.6.1	2020-04-29	[1]	CRAN (R 4.0.0)	
withr	2.2.0	2020-04-20	[2]	CRAN (R 4.0.0)	

```
[1] /Users/hofmann/Library/R/4.0/library
[2] /Library/Frameworks/R.framework/Versions/4.0/Resources/library
```

IS SHINY INSTALLED?

- Run the following command in your RStudio Console:

```
library(shiny)
```

- Check that you have a new(ish) version of shiny
1.5.0 is the newest version of shiny

Your Turn: check the version of your shiny package

- If either one of the above commands fails/or the answer is no:

```
install.packages("shiny")
```

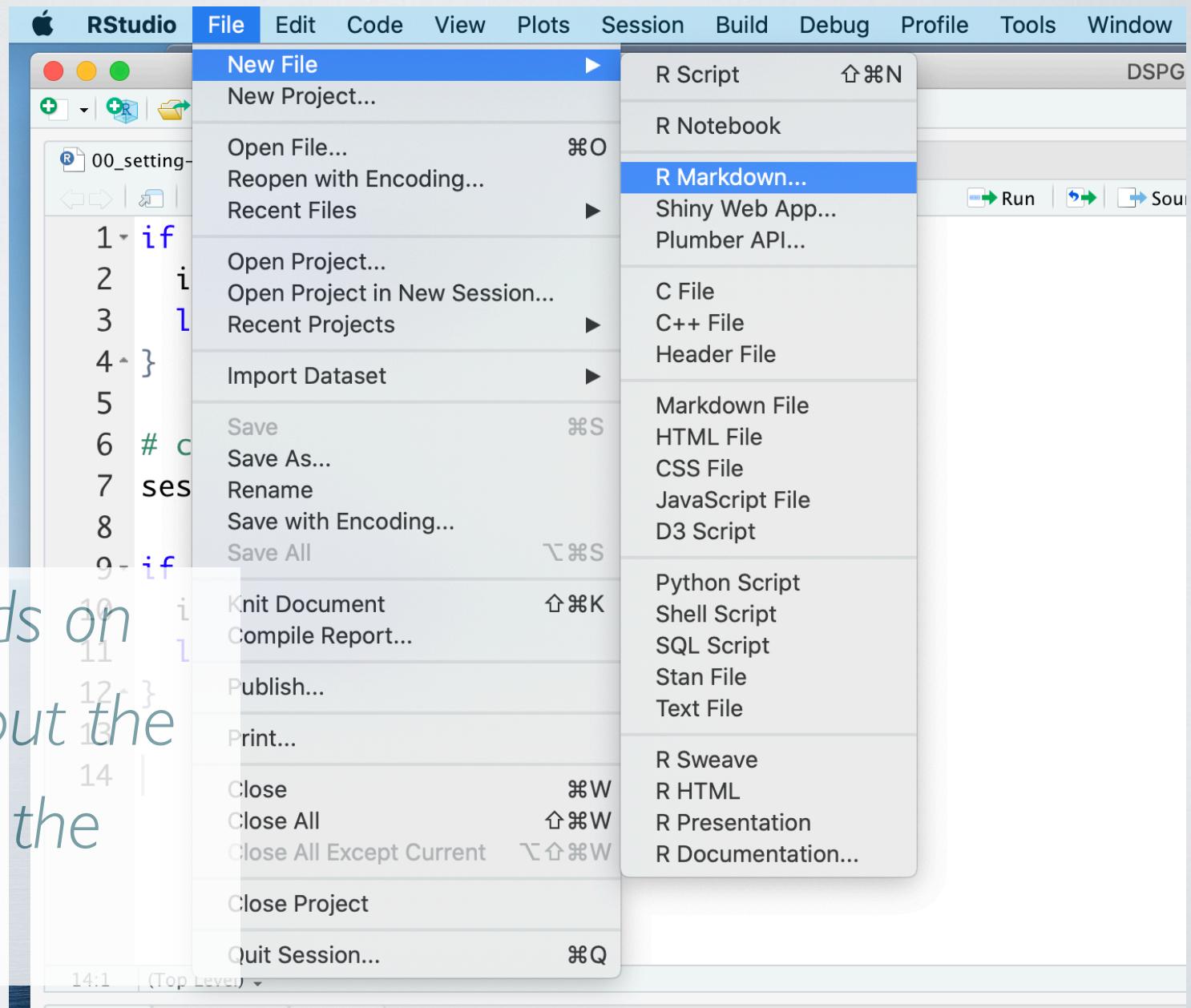


A FIRST APP

NEXT 5 MINS

- Create a new shiny app in RStudio
- Run it
- Get out of it

CREATE A NEW SHINY APP



*The UI depends on
the platform, but the
functionality is the
same*

DSPG - RStudio

File Edit View Insert Cell Help Addins

00_setting-up.R x

Source on Save | Run | Source | Environment History Connections Tutorial

Import Dataset | Global Environment

Environment is empty

1 if (!require(devtools)) {
2 install.packages("devtools")
3 library(devtools)
4 }
5
6 # check your session info:
7 session_info()
8
9 if (!require(shiny)) {
10 install.packages("shiny")
11 library(shiny)
12 }
13
14

14:1 (Top Level) ▾

Console Terminal Jobs

~/Documents/DSPG/DSPG/ ↵

Natural Language support but running in an English Locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/Documents/DSPG/DSPG/.RData]

> |

New Shiny Web Application

Application name: my-first-app

Application type: Single File (app.R) Multiple File (ui.R/server.R)

Create within directory: ~/Documents/DSPG/DSPG

Shiny Web Applications Create Cancel

Viewer

	Size	Modified
SPG	2.5 KB	Jun 27, 2020, 11:55 AM
	187 B	Jun 27, 2020, 11:55 AM
	193 B	Jun 27, 2020, 11:24 AM
	204 B	Jun 27, 2020, 11:55 AM

DSPG - RStudio

Environment History Connections Tutorial

Import Dataset Global Environment

Run App

00_setting-up.R app.R

This is a Shiny web application. You can run the application by c
the 'Run App' button above.

Find out more about building applications with Shiny here:

<http://shiny.rstudio.com/>

library(shiny)

Define UI for application that draws a histogram
ui <- fluidPage(

 # Application title
 titlePanel("Old Faithful Geyser Data"),

 # Sidebar with a slider input for number of bins
)

1:1 (Top Level) R Script

Console Terminal Jobs

~/Documents/DSPG/DSPG/ Natural Language support but running in an English Locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/Documents/DSPG/DSPG/.RData]

> |

Environment is empty

Files Plots Packages Help Viewer

New Folder Delete Rename More

Home > Documents > DSPG > DSPG

Name	Size	Modified
..	2.5 KB	Jun 27, 2020, 11:55 AM
.RData	187 B	Jun 27, 2020, 11:55 AM
.Rhistory	193 B	Jun 27, 2020, 11:24 AM
00_setting-up.R	204 B	Jun 27, 2020, 11:55 AM
DSPG.Rproj		
R code		
my-first-app		



DSPG - RStudio

00_setting-up.R x app.R x Go to file/function Addins Environment History Connections Tutorial DSPG

Run App Import Dataset Global Environment List

Environment is empty

1 #
2 # This is a Shiny web application. You can run the application by c
3 # the 'Run App' button above.
4 #
5 # Find out more about building applications with Shiny here:
6 #
7 # <http://shiny.rstudio.com/>
8 #
9
10 library(shiny)
11
12 # Define UI for application that draws a histogram
13 ui <- fluidPage(
14
15 # Application title
16 titlePanel("Old Faithful Geyser Data"),
17
18 # Sidebar with a slider input for number of bins
1:1 (Top Level) ▾ R Script ▾

Console Terminal x Jobs x ~/Documents/DSPG/DSPG/ Natural Language support but running in an English Locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/Documents/DSPG/DSPG/.RData]

> |

Files Plots Packages Help Viewer

New Folder Delete Rename More

Home > Documents > DSPG > DSPG

Name	Size	Modified
..	2.5 KB	Jun 27, 2020, 11:55 AM
.RData	187 B	Jun 27, 2020, 11:55 AM
.Rhistory	193 B	Jun 27, 2020, 11:24 AM
00_setting-up.R	204 B	Jun 27, 2020, 11:55 AM
DSPG.Rproj		
R code		
my-first-app		

Old Faithful Geyser Data

Number of bins:

Histogram of x

Frequency

x

This is how things should look like

Modified

Jun 27, 2020, 11:55 AM

Jun 27, 2020, 11:55 AM

Jun 27, 2020, 11:24 AM

Jun 27, 2020, 11:55 AM

DSPG - RStudio

00_setting-up.R x app.R x Addins ▾ DSPG ▾

Reload App ▾

This is a Shiny web application. You can run the application by clicking the 'Run App' button above.

Find out more about building applications with Shiny here:

<http://shiny.rstudio.com/>

library(shiny)

Define UI for application that draws a histogram

ui <- fluidPage(

Application title

titlePanel("Old Faithful Geyser Data"),

Sidebar with a slider input for number of bins

1:1 (Top Level) ▾

R Script ▾

Console Terminal Jobs

~/Documents/DSPG/DSPG/ ↵

'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or 'help.start()' for an HTML browser interface to help.

Type 'q()' to quit R.

[Workspace loaded from ~/Documents/DSPG/DSPG/.RData]

> shiny::runApp('my-first-app')

Loading required package: shiny

Listening on http://127.0.0.1:4127

Environment History Connections Tutorial

Import Dataset

Global Environment

Environment is empty

Files Plots Packages Help Viewer

New Folder Delete Rename More

Home > Documents > DSPG > DSPG

Size Modified

.Rhistory 2.5 KB Jun 27, 2020, 11:55 AM

00_setting-up.R 187 B Jun 27, 2020, 11:55 AM

DSPG.Rproj 193 B Jun 27, 2020, 11:24 AM

R code 204 B Jun 27, 2020, 11:55 AM

my-first-app

Looking like this?

A screenshot of the RStudio interface. A large orange arrow points from the text "Sometimes the shiny window hides behind the RStudio Viewer" towards the top menu bar. The menu bar shows "Window" is selected, with "Minimize", "Zoom", and "Bring All to Front" options. The "Bring All to Front" option is highlighted with a blue selection bar. The main workspace shows an R script with code for a Shiny app. The RStudio Viewer panel is visible on the right, showing a file tree and some environment information. The text "Environment is empty" is displayed in the viewer.

Sometimes the shiny window
hides behind the
RStudio Viewer

```
1 #  
2 # This is a Shiny web application. You can run  
3 # the 'Run App' button above.  
4 #  
5 # Find out more about building applications with Shiny here:  
6 #  
7 # http://shiny.rstudio.com/  
8 #  
9  
10 library(shiny)  
11  
12 # Define UI for application that draws a histogram  
13 ui <- fluidPage(  
14  
15   # Application title  
16   titlePanel("Old Faithful Geyser Data"),  
17  
18   # Sidebar with a slider input for number of bins
```

1:1 (Top Level) R Script

Console Terminal Jobs

~/Documents/DSPG/DSPG/ ↵

'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/Documents/DSPG/DSPG/.RData]

```
> shiny::runApp('my-first-app')
Loading required package: shiny
```

Listening on http://127.0.0.1:4127

```
00_setting-up.R x app.R x
1 #  
2 # This is a Shiny web application.  
3 # the 'Run App' button above.  
4 #  
5 # Find out more about building apps  
6 #  
7 # http://shiny.rstudio.com/  
8 #  
9  
10 library(shiny)  
11  
12 # Define UI for application that  
13 ui <- fluidPage(  
14  
15     # Application title  
16     titlePanel("Old Faithful Geyser Data")  
17  
18     # Sidebar with a slider input  
19     sidebarLayout(
```

1:1 (Top Level) ▾

Console Terminal Jobs

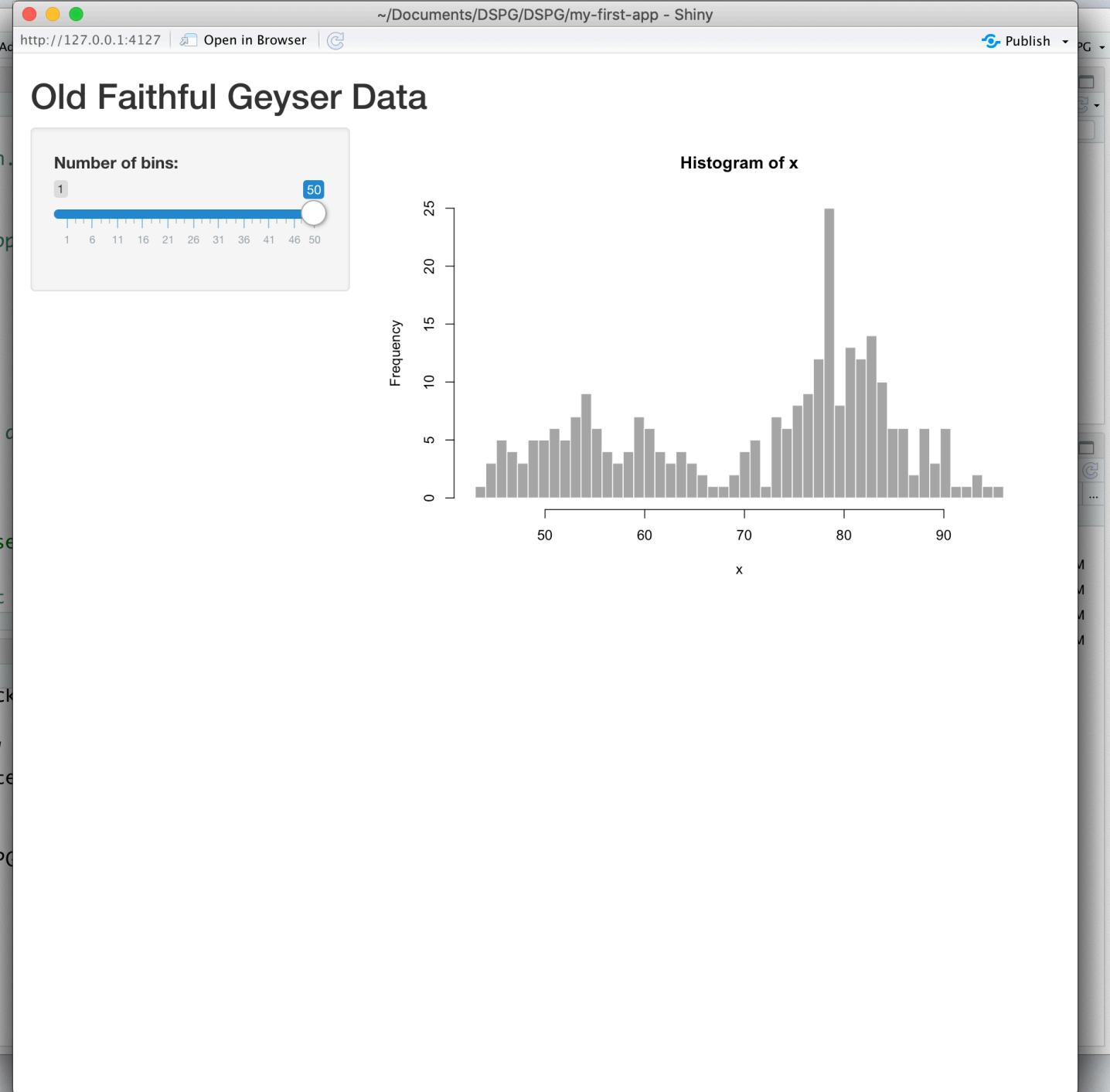
~/Documents/DSPG/DSPG/ ↗

'citation()' on how to cite R or R packages.

Type 'demo()' for some demos, 'help()' or 'help.start()' for an HTML browser interface. Type 'q()' to quit R.

Now interact with
the app

Listening on http://127.0.0.1:4127



DSPG - RStudio

00_setting-up.R x app.R x Addins ▾ DSPG ▾

1 #
2 # This is a Shiny web application. You can run the application by c
3 # the 'Run App' button above.
4 #
5 # Find out more about building applications with Shiny here:
6 #
7 # <http://shiny.rstudio.com/>
8 #
9
10 library(shiny)
11
12 # Define UI for application that draws a histogram
13 ui <- fluidPage(
14 # ...
15)

Make sure to stop the app once we are done interacting with it

Type 'q()' to quit R.

[Workspace loaded from ~/Documents/DSPG/DSPG/.RData]

> shiny::runApp('my-first-app')
Loading required package: shiny

Listening on http://127.0.0.1:4127

Environment History Connections Tutorial

Import Dataset

Global Environment

Environment is empty

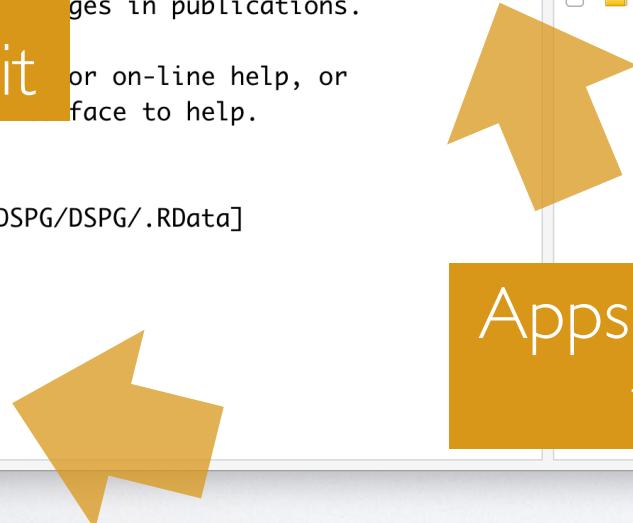
Files Plots Packages Help Viewer

New Folder Delete Rename More

Home > Documents > DSPG > DSPG

Name	Size	Modified
..	2.5 KB	Jun 27, 2020, 11:55 AM
.RData	187 B	Jun 27, 2020, 11:55 AM
.Rhistory	193 B	Jun 27, 2020, 11:24 AM
00_setting-up.R	204 B	Jun 27, 2020, 11:55 AM
DSPG.Rproj		
R code		
my-first-app		

Apps take over the control of the RStudio viewer





HOW DOES SHINY WORK?

TWO MAIN PARTS

- What we see and interact with:
 - The user interface (UI):
layout with user input and (plot) output
- What is going on underneath:
 - The server: the glue between input and output

MINIMAL EXAMPLE

```
library(shiny)
```

```
ui <- fluidPage(
```

The user interface

```
)
```

```
server <- function(input, output, session) {
```

```
}
```

```
shinyApp(ui, server)
```

The glue between
input and output

Running the app

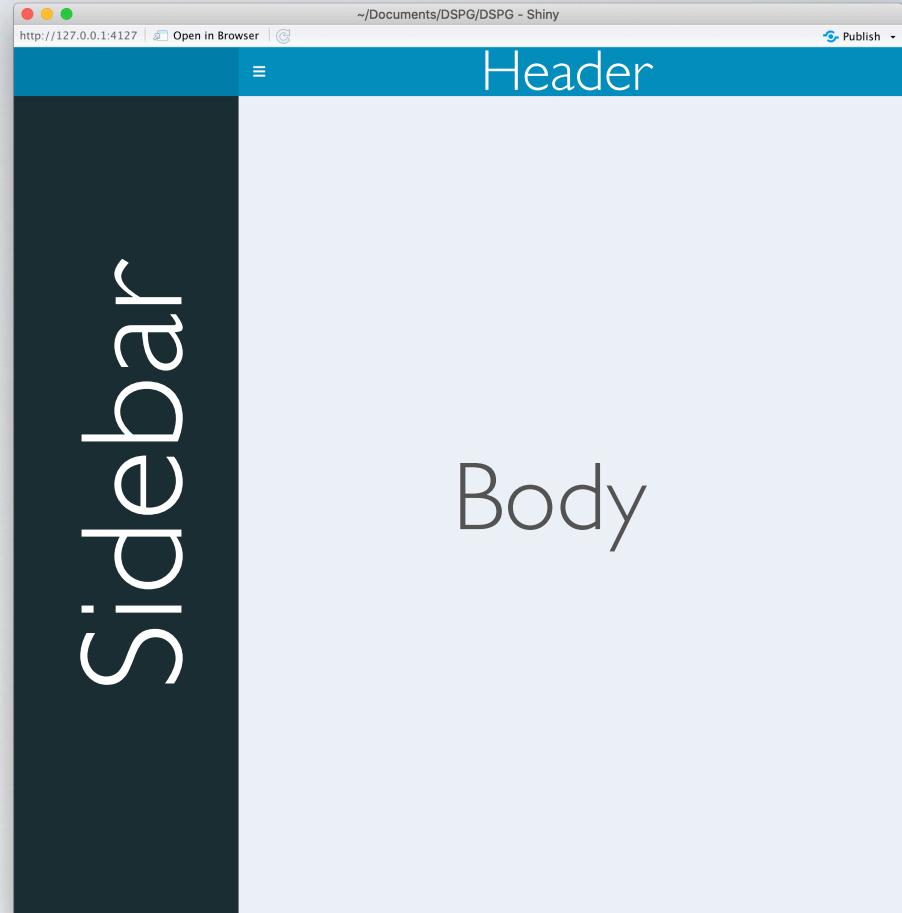
```
library(shiny)
library(shinydashboard)

ui <- dashboardPage(
  dashboardHeader(),
  dashboardSidebar(),
  dashboardBody()
)

server <- function(input, output, session) {

}

shinyApp(ui, server)
```



```
library(shiny)
library(shinydashboard)

sidebar <- dashboardSidebar(
 textInput("name",
    "Enter your name:",
    value = "Heike")
)

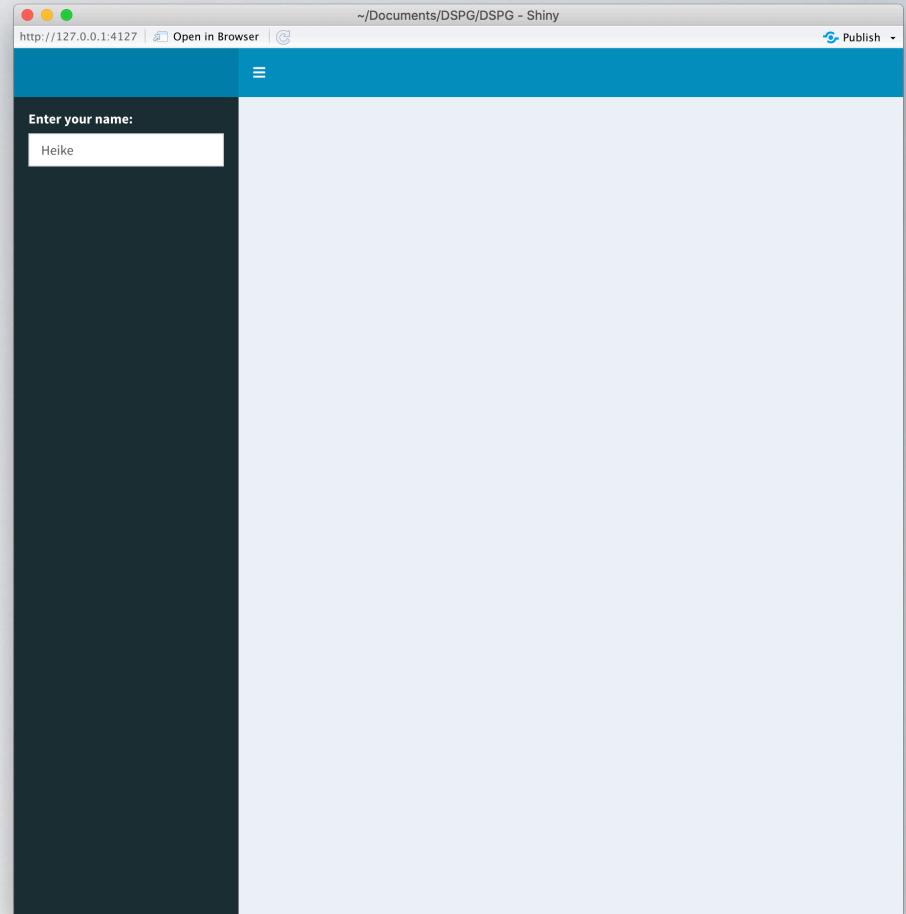
ui <- dashboardPage(
  dashboardHeader(),
  sidebar = sidebar,
  dashboardBody( )
)

server <- function(input, output, session) {

}

shinyApp(ui, server)

server <- function(input, output, session) {
```



INPUT WIDGETS

http://127.0.0.1:3771 | [Open in Browser](#) |

Publish ▾

Basic widgets

Buttons

Action

Submit

Date range

2017-06-21 to 2017-06-21

Single checkbox

Choice A

Checkbox group

- Choice 1
- Choice 2
- Choice 3

Date input

2014-01-01

Radio buttons

- Choice 1
- Choice 2
- Choice 3

Select box

Choice 1 ▾

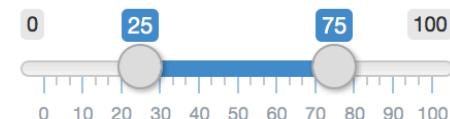
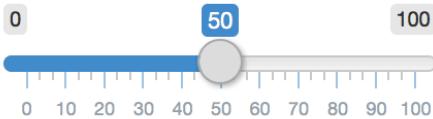
Help text

Note: help text isn't a true widget, but it provides an easy way to add text to accompany other widgets.

Numeric input

1

Sliders



Text input

Enter text...

INPUT WIDGETS

- Each input widget corresponds to a function of the form
`xxxInput(inputId, label, value, ...)`
- **inputId:** internal name of the widget
label: text shown to user
value: default value at the start
...: other widget specific parameters

<https://shiny.rstudio.com/images/shiny-cheatsheet.pdf>

Inputs - collect values from the user

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

Action

`actionButton(inputId, label, icon, ...)`

Link

`actionLink(inputId, label, icon, ...)`

Choice 1

Choice 2

Choice 3

Check me

`checkboxGroupInput(inputId, label, choices, selected, inline)`

`checkboxInput(inputId, label, value)`



`dateInput(inputId, label, value, min, max, format, startview, weekstart, language)`



`dateRangeInput(inputId, label, start, end, min, max, format, startview, weekstart, language, separator)`

Choose File

`fileInput(inputId, label, multiple, accept)`

1

`numericInput(inputId, label, value, min, max, step)`

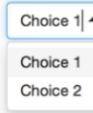
.....

`passwordInput(inputId, label, value)`

Choice A

Choice B

Choice C



`radioButtons(inputId, label, choices, selected, inline)`

`selectInput(inputId, label, choices, selected, multiple, selectize, width, size) (also selectizeInput())`



`sliderInput(inputId, label, min, max, value, step, round, format, locale, ticks, animate, width, sep, pre, post)`

Apply Changes

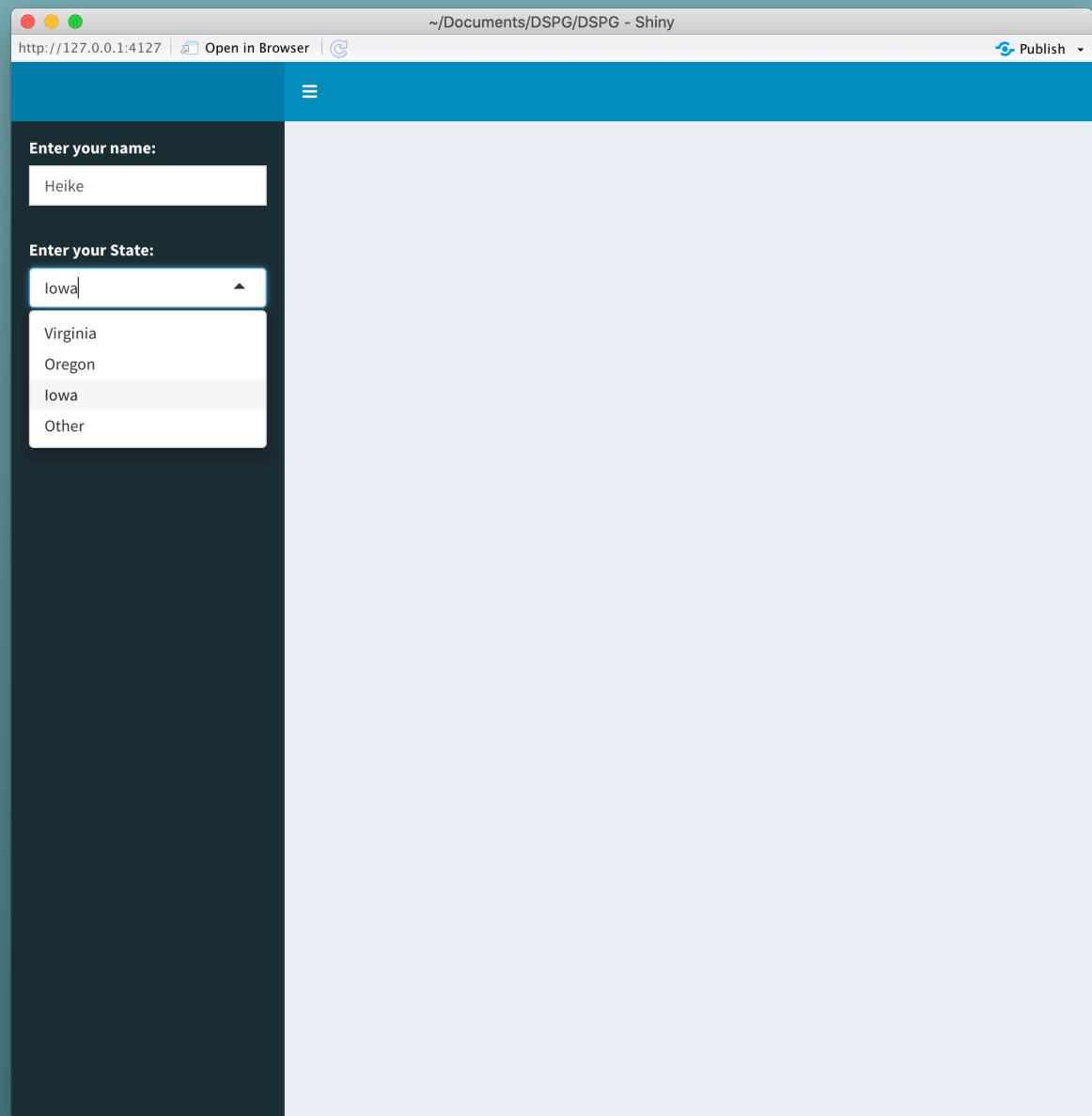
`submitButton(text, icon)`
(Prevents reactions across entire app)

Enter text

`textInput(inputId, label, value)`

YOUR TURN

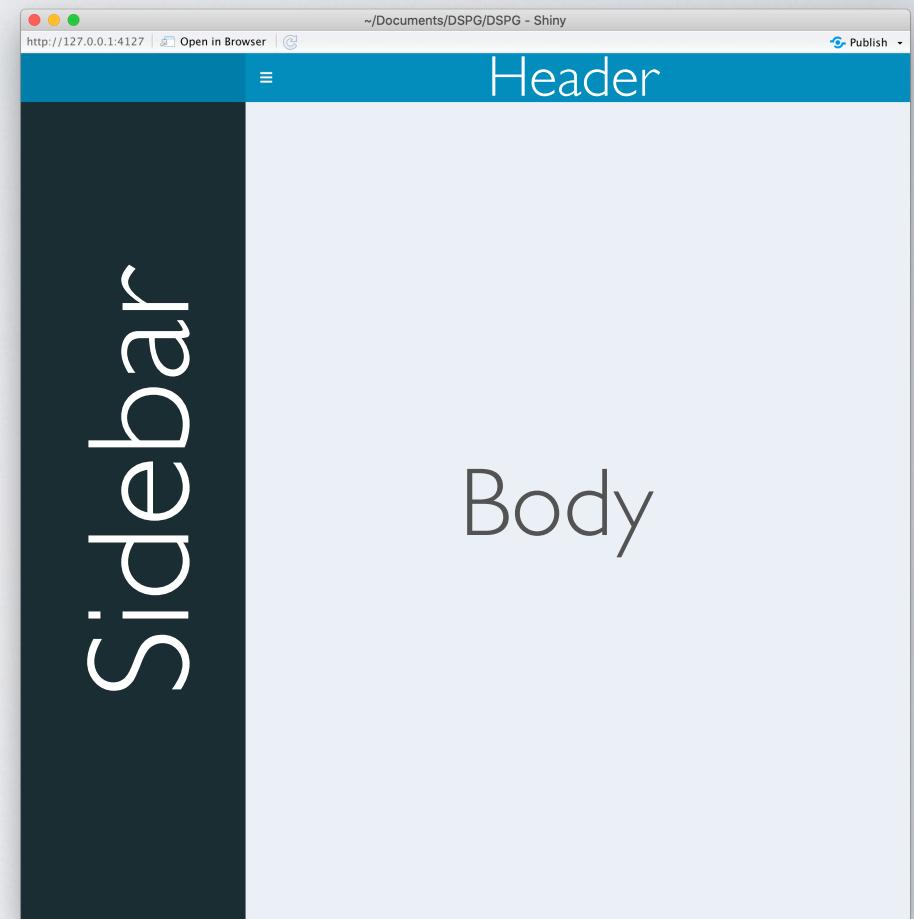
- Open the file app.R in folder 02b_minimal
- Modify the function sidebar to include an input as shown on the right
- Advanced: give the app a title
- The answer is in 02c_minimal



ADDING OUTPUT

- Output typically goes into the body of an app
- Output functions have the form

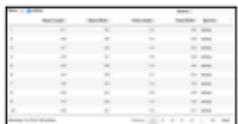
`xxxOutput(outputid)`



- Output can be in form of tables, plots, text, ...

RENDERING OUTPUT

Outputs - render*() and *Output() functions work together to add R output to the UI



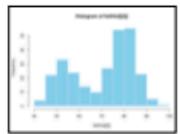
DT::renderDataTable(expr, options, callback, escape, env, quoted)



dataTableOutput(outputId, icon, ...)



renderImage(expr, env, quoted, deleteFile)



renderPlot(expr, width, height, res, ..., env, quoted, func)

'data.Frame': 3 obs. of 2 variables:
\$ Sepal.Length: num 5.1 4.9 6.7
\$ Sepal.Width : num 3.5 3 3.2

renderPrint(expr, env, quoted, func, width)

imageOutput(outputId, width, height, click, dblclick, hover, hoverDelay, hoverDelayType, brush, clickId, hoverId, inline)

plotOutput(outputId, width, height, click, dblclick, hover, hoverDelay, hoverDelayType, brush, clickId, hoverId, inline)

verbatimTextOutput(outputId)

Usually in
tableOutput(outputId)
dashboardBody()

textOutput(outputId, container, inline)

A screenshot of a rendered table in a browser window. The table has columns for Sepal.Length, Sepal.Width, Petal.Length, Petal.Width, and Species.

foo

renderText(expr, env, quoted, func)

uiOutput(outputId, inline, container, ...)

& **htmlOutput(outputId, inline, container, ...)**



renderUI(expr, env, quoted, func)

```

library(shiny)
library(shinydashboard)
library(ggplot2)

sidebar <- dashboardSidebar(
 textInput("name", "Enter your name"),
  selectInput("state", "Enter your State",
    choices=c("Virginia",
              "selected = "Iowa"))
)

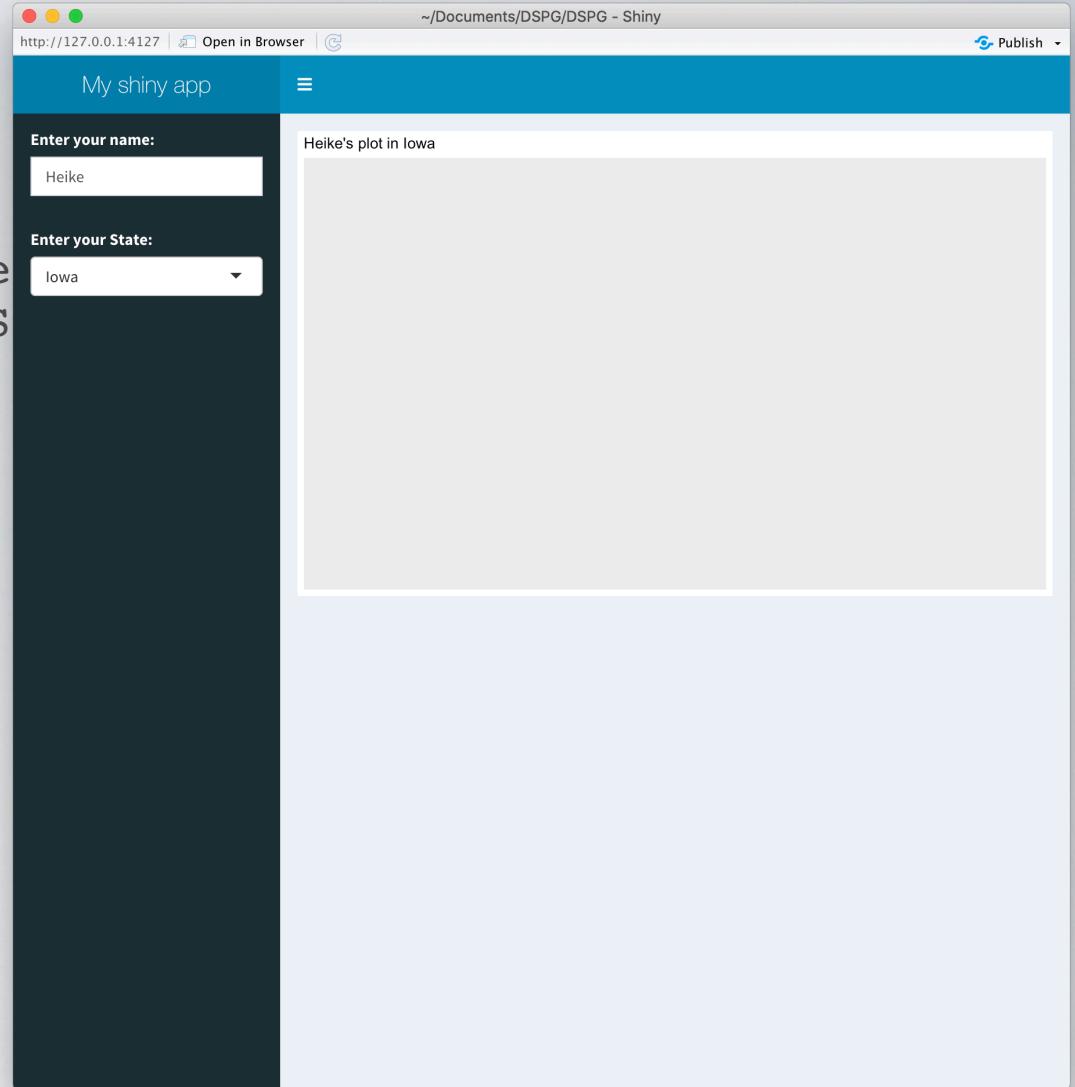
body <- dashboardBody(
  plotOutput("myplot")
)

ui <- dashboardPage(
  dashboardHeader(title = "My shiny app"),
  sidebar = sidebar,
  body = body
)

server <- function(input, output, session) {
  output$myplot <- renderPlot({
    ggplot() +
      ggttitle(sprintf("%s's plot in %s", input$name, input$state))
  })
}

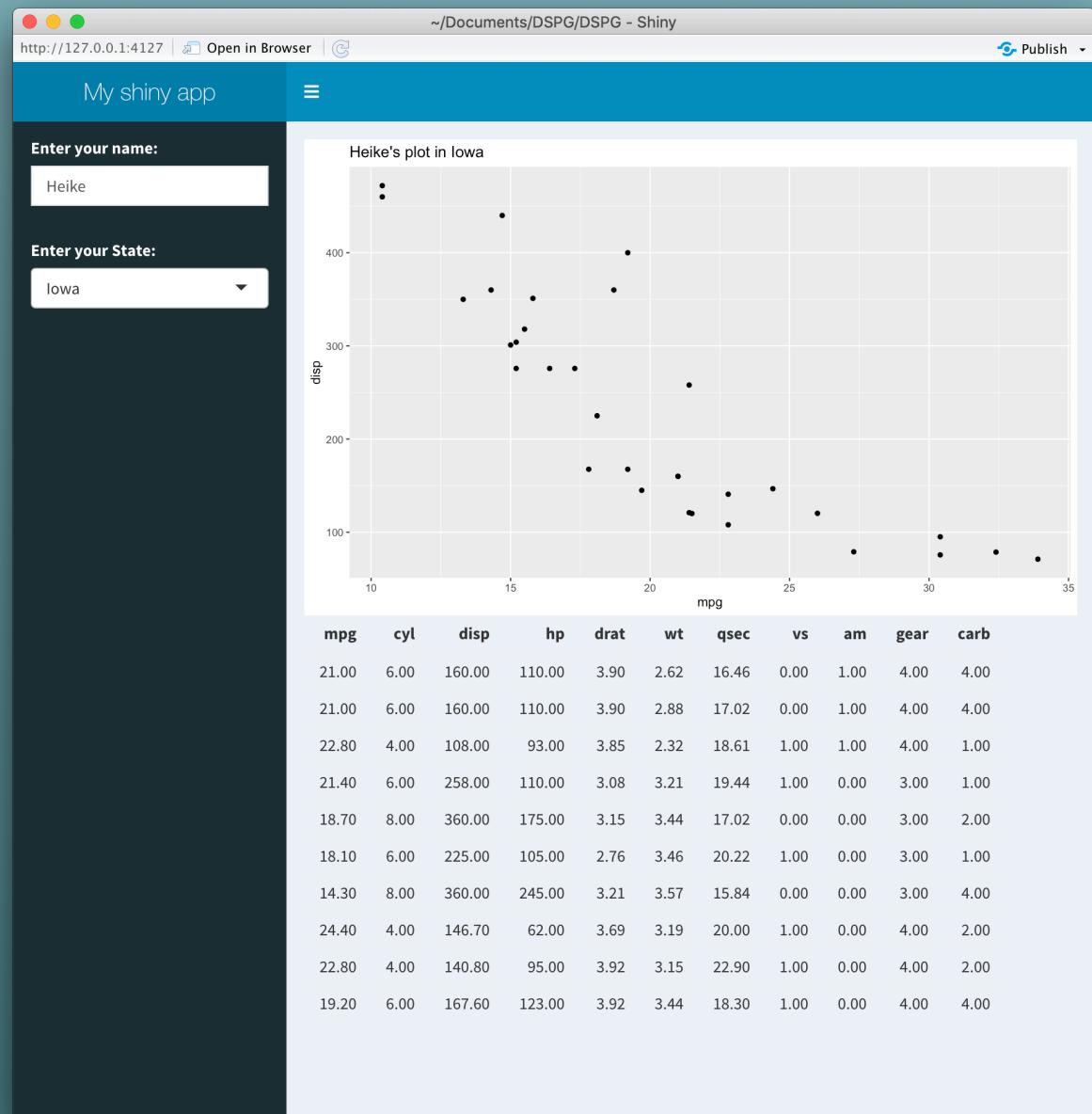
shinyApp(ui, server)

```



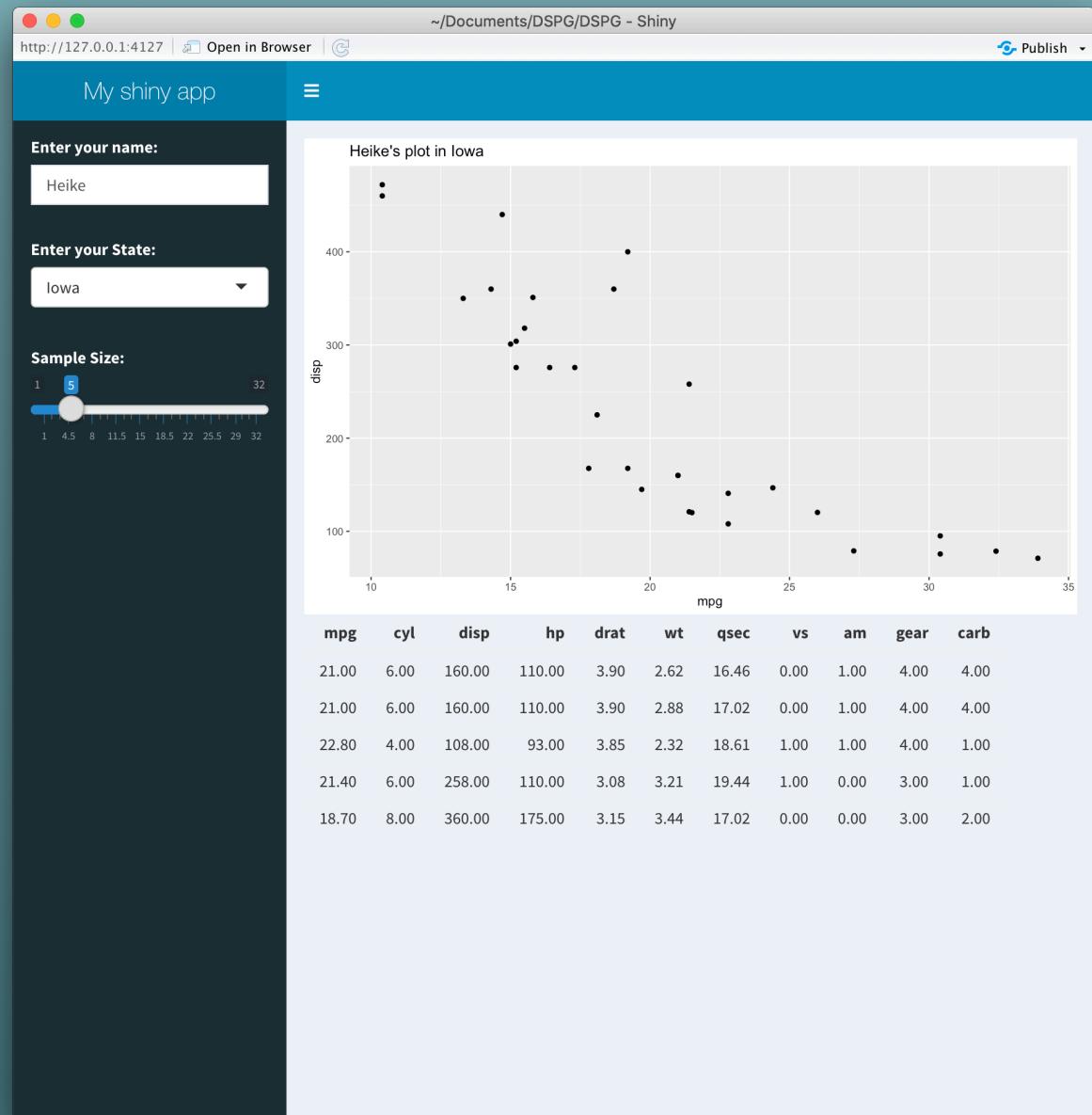
YOUR TURN

- Open the file app.R in folder 02d_minimal
- Modify the plot to show data points from the mtcars data
- Add a table below to show the first ten lines of the data
- The answer is in 02e_minimal



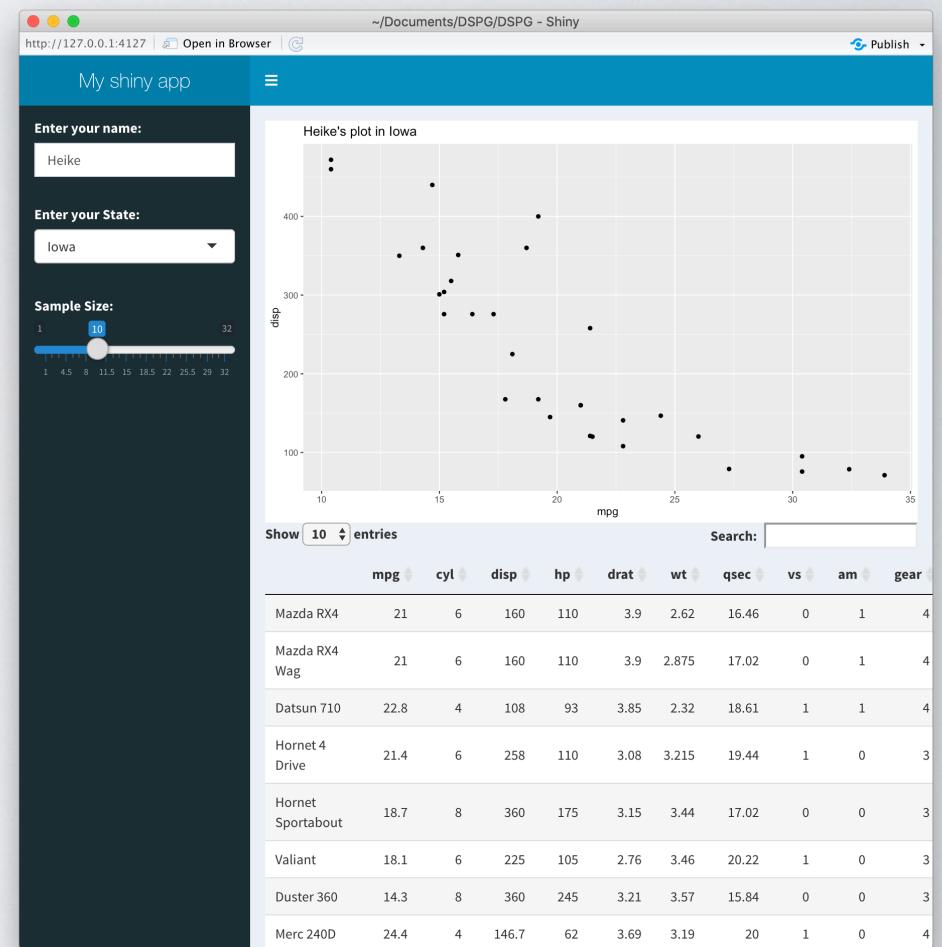
YOUR TURN

- Open the file app.R in folder 02e_minimal
 - Include a slider with values between 1 and 32
 - Make the number of data rows shown in the table dependent on the slider
 - The answer is in 02f_minimal



FANCIER OUTPUTS

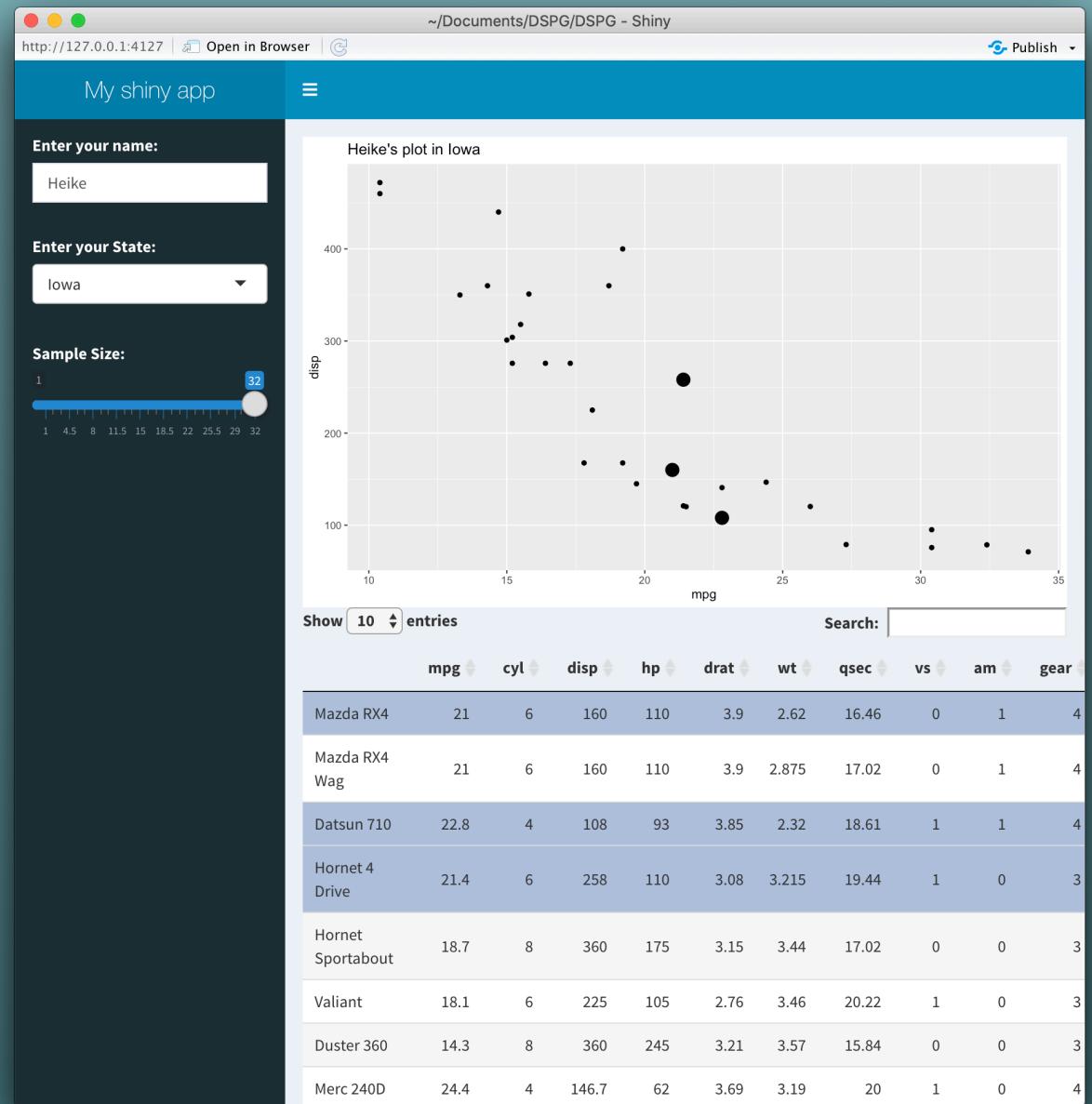
- The package DT provides functions to render data tables
- DTOutput and renderDT help defining and rendering the table
- See 02g_minimal.R



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

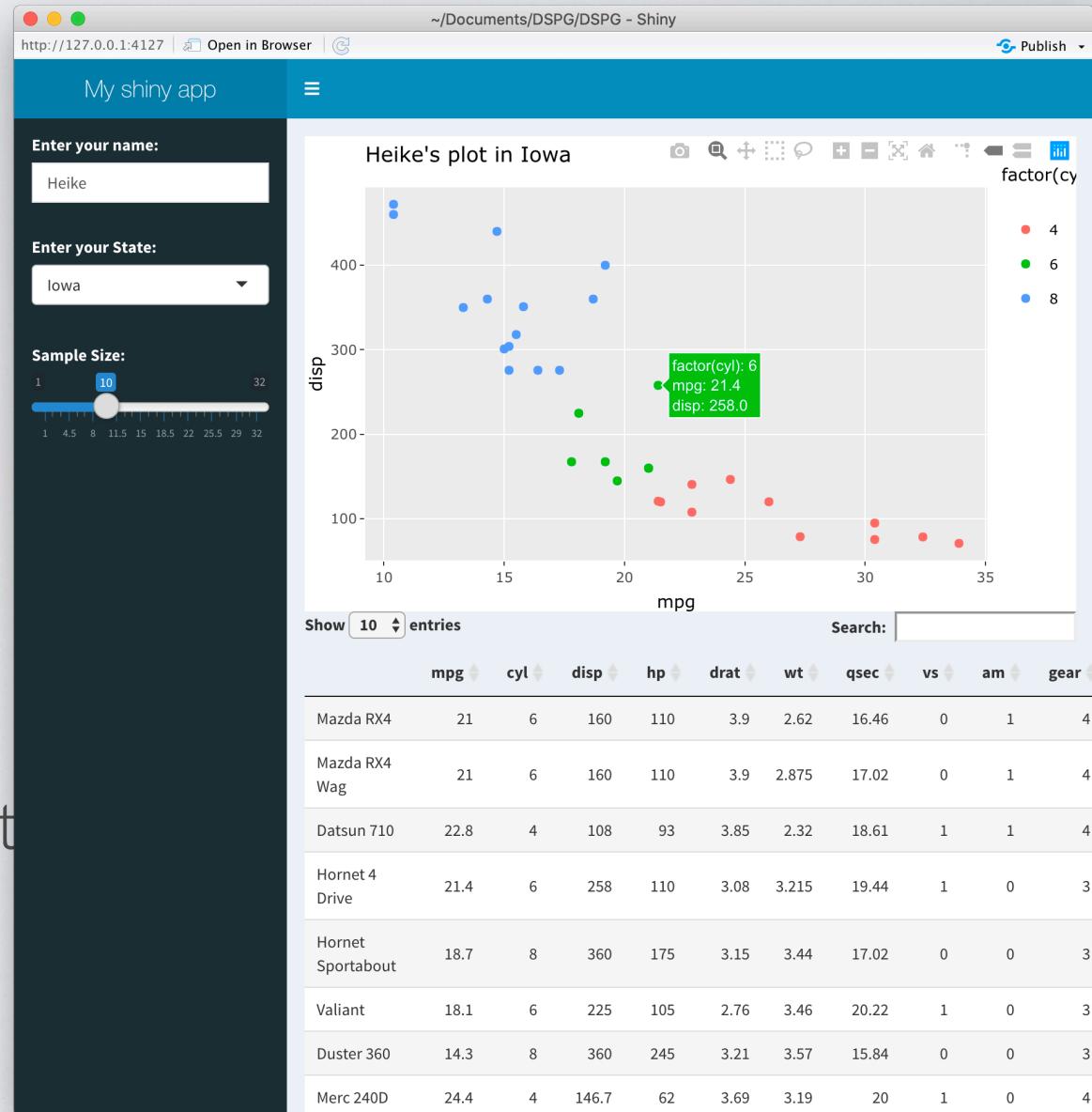
YOUR TURN

- Open the file app.R in folder 02h_minimal
 - Make points bigger in the plot that are selected in the table
 - `input$tableId_rows_selected` contains all of the ids of the selected rows
 - The answer is in 02i_minimal



FANCIER OUTPUTS

- The package `plotly` implements interactive plots
try `plotly(gg)` of a `ggplot2` plot
 - `plotlyOutput` and `renderPlotly` define and render the interactive plot
 - See `02j_minimal.R`



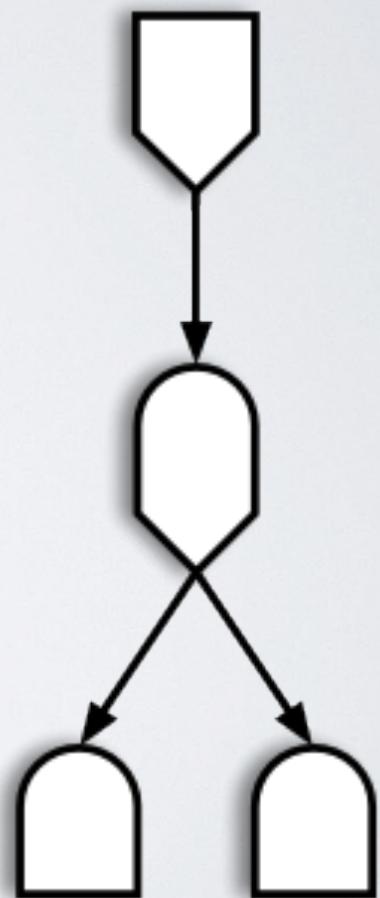
QUESTIONS?



REACTIVITY

ELEMENTS OF REACTIVITY

- Sources
 - Any input widget is a source
- Conductors
 - Use input and are being used further along
- Observers
 - Any output is an observer



TWO CONDUCTORS

- Reactive expressions and reactive events are two types of conductors
- Reactive expressions are the archetypical conductor: envelope functionality used in multiple places of an app, run evaluations only once and store current values.
- Reactive events are only triggered by specific events (such as a click on an action button)

REACTIVE EXPRESSIONS

```
rval <- reactive({  
...  
})
```

Called like a function as:
`rval()`

- reactive expressions are executed **lazily**, and their values are **cached**
- **Lazy:** evaluated only on demand, typically requested by a reactive endpoint.
- **Cached:** (re-)evaluated only when the value of a dependency changed.

REACTIVE EVENTS

```
rval <- eventReactive(actionbutton, {  
...  
})
```

Called like a function as:
`rval()`

- reactive events are executed even more **lazily**: only on demand, typically requested by an action button

EXAMPLE: SUBMISSION FORM

- In RStudio, open file 03_submission.R
- Run the app (a couple of times)
- Turn on showcase mode

The screenshot shows a Shiny application running in a web browser. The title bar indicates the application is titled '~/Documents/DSPG/DSPG - Shiny' and the URL is 'http://127.0.0.1:4127'. The browser has standard OS X-style window controls (red, yellow, green) in the top-left corner.

The application interface is a 'Submission Form' with the following fields:

- A text input field labeled 'Enter your Name'.
- A date input field labeled 'Enter your favorite day of the year' containing the value '2020-06-28'.
- A 'Submit' button at the bottom right.

On the far right of the browser window, there is a 'Publish' dropdown menu.

03_SUBMISSION.R

```
library(shiny)
library(lubridate) # install.packages("lubridate")
library(tidyverse)

ui <- fluidPage(
  h3("Submission Form"),
 textInput("name", "Enter your Name"),
  dateInput("date", "Enter your favorite day of the year", value=today()),
  actionButton("submit", "Submit"),
  h2(textOutput("greeting")),
  plotOutput("histogram")
)

server <- function(input, output, session) {
  rval_favs <- eventReactive(input$submit, {
    if (!file.exists("favorite-days.csv")) {
      favs <- data.frame(name=input$name, date = input$date, stringsAsFactors = FALSE)
    } else {
      favs <- read.csv("favorite-days.csv", stringsAsFactors = FALSE)
      favs <- rbind(favs, data.frame(name=input$name,
                                      date = as.character(input$date),
                                      stringsAsFactors = FALSE))
    }
    favs <- favs %>% mutate(date = lubridate::ymd(date))
    write.csv(favs, "favorite-days.csv", row.names=FALSE)
  })
}
```

03_SUBMISSION.R

```
library(shiny)
library(lubridate) # install.packages("lubridate")
library(tidyverse)

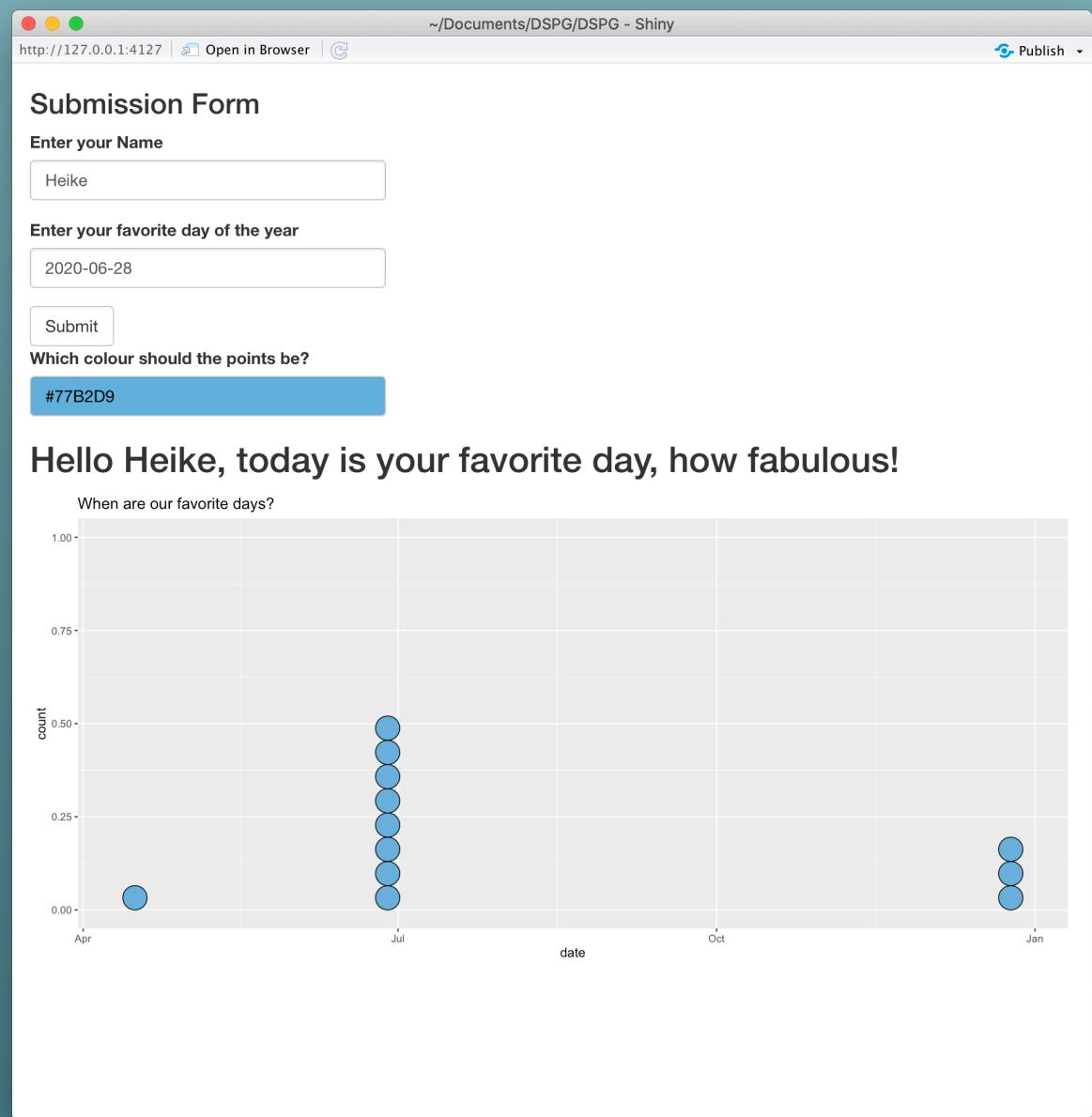
ui <- fluidPage(
  h3("Submission Form"),
 textInput("name", "Enter your Name"),
  dateInput("date", "Enter your favorite day of the year", value=today()),
  actionButton("submit", "Submit"),
  h2(textOutput("greeting")),
  plotOutput("histogram")
)

server <- function(input, output, session) {
  output$greeting <- eventReactive(input$submit, {
    daysdiff <- today() - input$date
    if (daysdiff > 1) dateout <- paste0("it has been ", daysdiff, " days since your favorite day.")
    else if (daysdiff == 1) dateout <- paste0("yesterday was your favorite day, I hope you enjoyed it!")
    else if (daysdiff == 0) dateout <- paste0("today is your favorite day, how fabulous!")
    else if (daysdiff == -1) dateout <- paste0("tomorrow is your favorite day, how exciting!")
    else dateout <- paste0("it is ", -daysdiff, " days until your favorite day.")

    paste0("Hello ", input$name, ", ", dateout)
  })
}
```

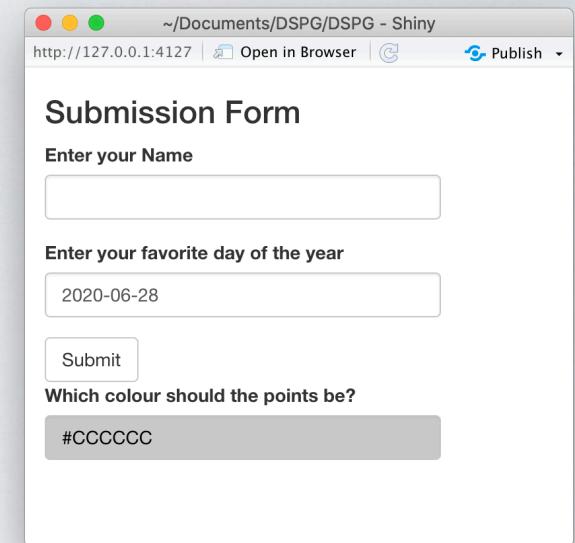
YOUR TURN

- Open the file 03_submission.R
- The package colourpicker implements a color wheel as input widget
- Allow users to change the color of the dots in the dot plot
- What other interactive elements can you think of adding?
- The answer is in 03b_submission.R



CONDITIONAL PANELS

- Showing a color picker before needed ... might be confusing users of the app
- `conditionalPanel(condition, ...)` allows us to encapsulate elements of the user interface and only show when 'condition' is fulfilled
- Here, a condition of
`condition = "input.submit > 0"`
is true when the submit button was pressed at least once
- This is implemented in 03c_submission.R



~/Documents/DSPG/DSPG - Shiny
http://127.0.0.1:4127 | Open in Browser | Publish

Submission Form

Enter your Name

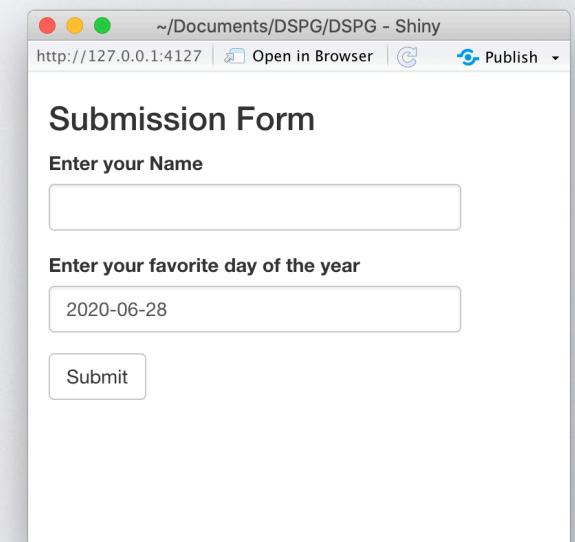
Enter your favorite day of the year

2020-06-28

Submit

Which colour should the points be?

#CCCCCC



~/Documents/DSPG/DSPG - Shiny
http://127.0.0.1:4127 | Open in Browser | Publish

Submission Form

Enter your Name

Enter your favorite day of the year

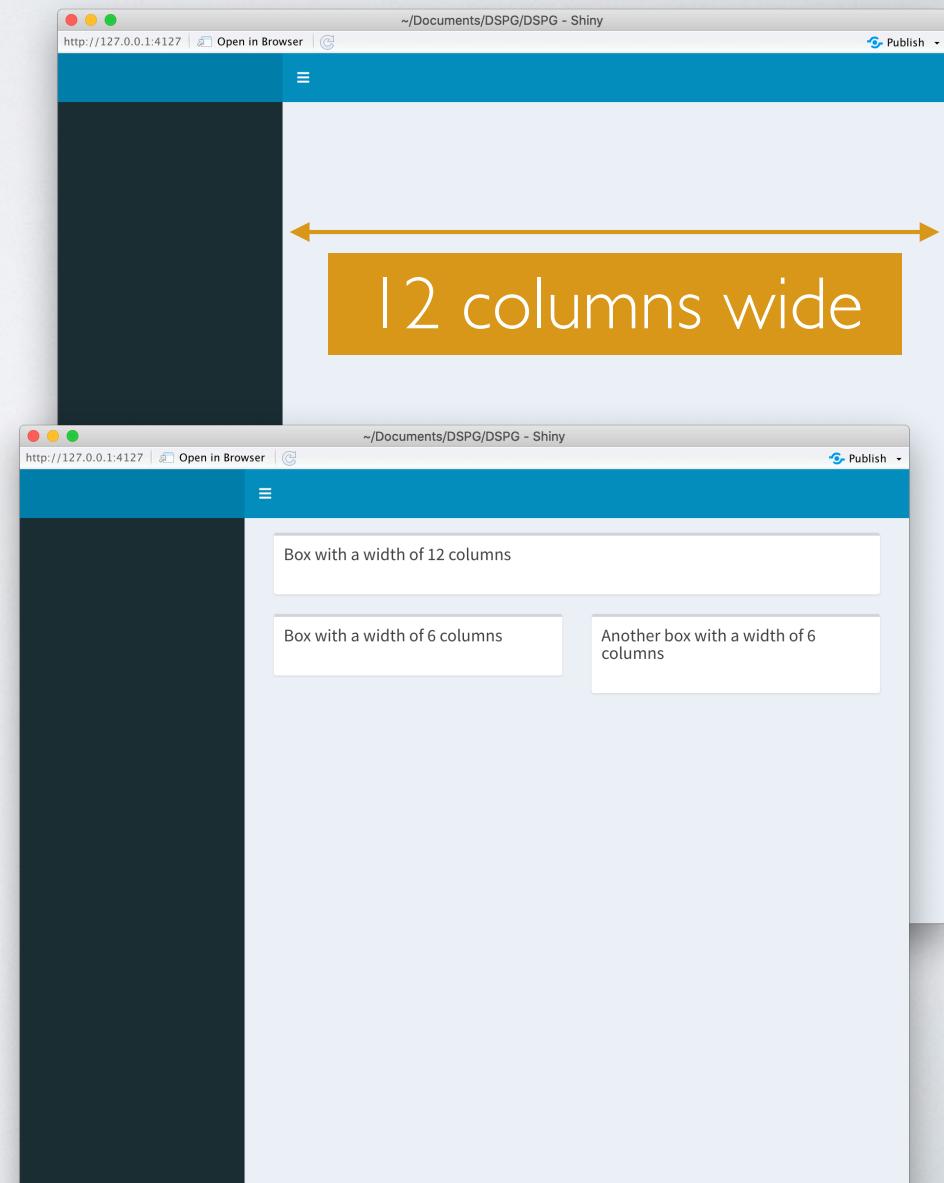
2020-06-28

Submit

LAYOUT OF DASHBOARDS

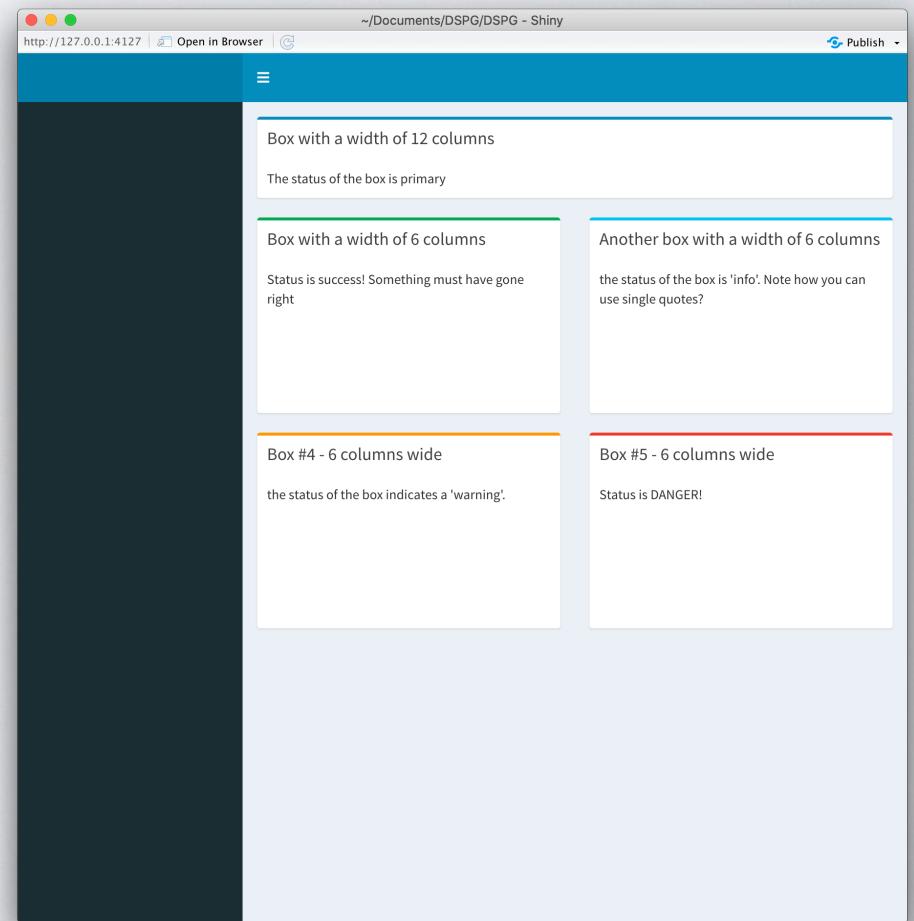
- The body can be laid out in a grid - either row based or column based
- Structure is introduced by boxes:

```
box(..., title = NULL,  
width = 6, height = NULL)
```



BOXES

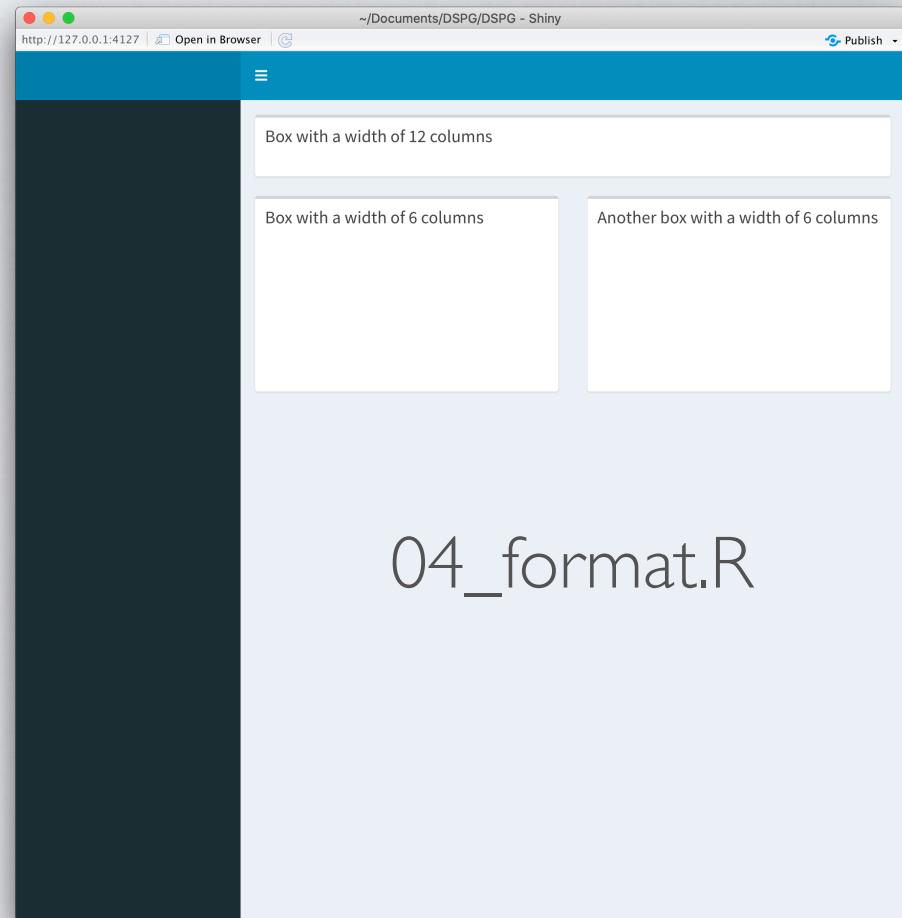
- Boxes help with structuring output
- Boxes also have a **status** parameter
- Status is shown as a colored bar along the top of a box
- **?validStatuses** are **primary**, **success**, **info**, **warning**, **danger**



ROW BASED LAYOUT

- Body is wrapped in a fluidRow function
- Tops of boxes are aligned
- Bottom of the boxes can be aligned by setting the height (in pixel)

```
body <- dashboardBody(  
  fluidRow(  
    box(title = "Box with a width of 12 columns", width = 12),  
    box(title = "Box with a width of 6 columns", width = 6, height = 200),  
    box(title = "Another box with a width of 6 columns", width = 6, height = 200)  
  )  
)
```



04_format.R

OTHER LAYOUTS

- In **column** based layouts, the body is wrapped in a fluidRow function
 - Height of boxes are aligned, each column has to define a width, boxes are aligned in width.
- In mixed layouts fluidRow and column can be used sequentially

TABS IN DASHBOARDS

The left screenshot shows a Shiny dashboard titled "My really complex app". The sidebar has two items: "Dashboard" and "Cars". The main content area is titled "Dashboard tab content" and contains four boxes:

- Box with a width of 12 columns: "Box with a width of 12 columns" and "The status of the box is primary".
- Box with a width of 6 columns: "Box with a width of 6 columns" and "Status is success! Something must have gone right".
- Another box with a width of 6 columns: "Another box with a width of 6 columns" and "the status of the box is 'info'. Note how you can use single quotes?".
- Box #4 - 6 columns wide: "Box #4 - 6 columns wide" and "the status of the box indicates a 'warning'".
- Box #5 - 6 columns wide: "Box #5 - 6 columns wide" and "Status is DANGER!".

The right screenshot shows the same dashboard with the "Cars" tab selected. The main content area is titled "What do you want to know about Cars?" and displays a scatter plot of "disp" vs "mpg" and a data table for the "Cars" dataset.

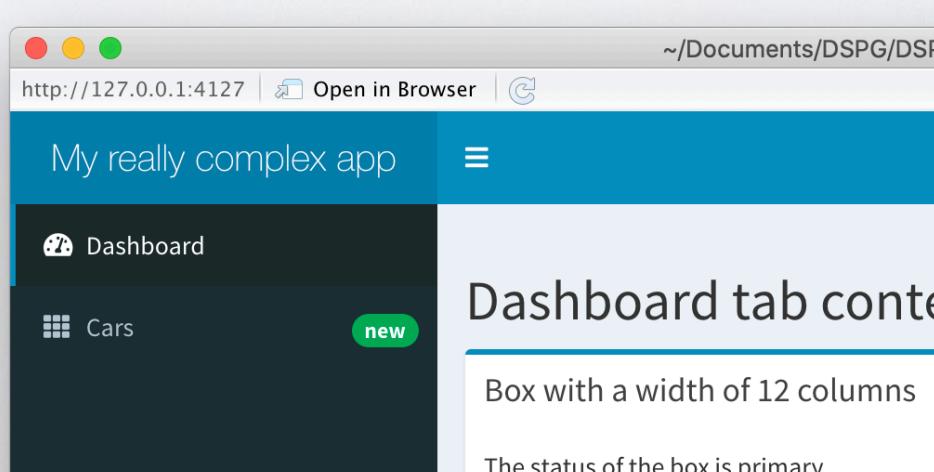
mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	
Mazda RX4	21	6	160	110	3.9	2.62	16.46	0	1	4
Mazda RX4 Wag	21	6	160	110	3.9	2.875	17.02	0	1	4
Datsun 710	22.8	4	108	93	3.85	2.32	18.61	1	1	4
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3
Hornet Sportabout	18.7	8	360	175	3.15	3.44	17.02	0	0	3
Valiant	18.1	6	225	105	2.76	3.46	20.22	1	0	3

- The sidebar menu can be used to introduce tabs for quick navigation

05_tabssets.R

TABS IN DASHBOARDS

```
sidebar <- dashboardSidebar(  
  sidebarMenu(  
    menuItem("Dashboard", tabName = "dashboard",  
            icon = icon("dashboard")),  
    menuItem("Cars", icon = icon("th"), tabName = "cars",  
            badgeLabel = "new", badgeColor = "green")  
)  
)
```



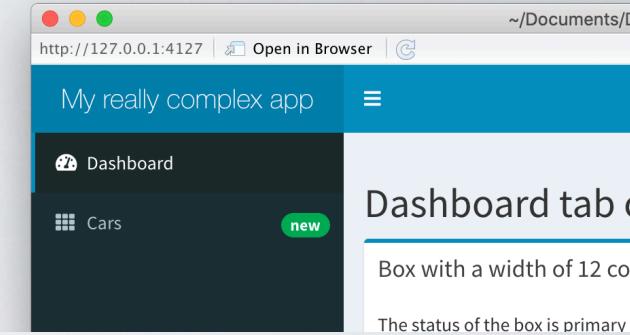
TABS IN DASHBOARDS

```

sidebar <- dashboardSidebar(
  sidebarMenu(
    menuItem("Dashboard", tabName = "dashboard",
              icon = icon("dashboard")),
    menuItem("Cars", icon = icon("th"), tabName = "cars",
              badgeLabel = "new", badgeColor = "green")
  )
)

body <- dashboardBody(
  tabItems(
    tabItem(tabName = "dashboard",
            h2("Dashboard tab content"),
            fluidRow(
              box(title = "Box with a width of 12 columns", width = 12,
                  status = "primary", "The status of the box is primary"),
              ...
              box(title = "Box #5 - 6 columns wide",
                  status = "danger", "Status is DANGER!",
                  width = 6, height = 200)
            )
    ),
    tabItem(tabName = "cars",
            h2("What do you want to know about Cars?"),
            plotOutput("myplot"),
            DTOutput("mytable"))
  )
)

```



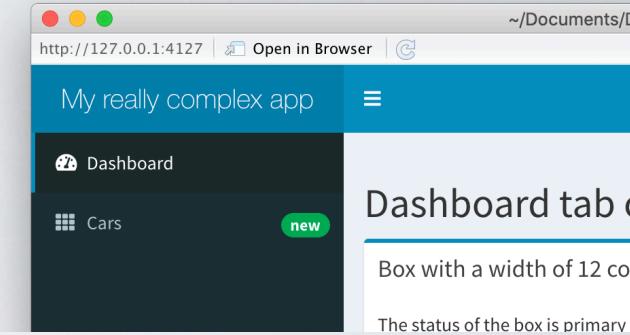
TABS IN DASHBOARDS

```

sidebar <- dashboardSidebar(
  sidebarMenu(
    menuItem("Dashboard", tabName = "dashboard",
              icon = icon("dashboard")),
    menuItem("Cars", icon = icon("th"), tabName = "cars",
              badgeLabel = "new", badgeColor = "green")
  )
)

body <- dashboardBody(
  tabItems(
    tabItem(tabName = "dashboard",
            h2("Dashboard tab content"),
            fluidRow(
              box(title = "Box with a width of 12 columns", width = 12,
                  status = "primary", "The status of the box is primary"),
              ...
              box(title = "Box #5 - 6 columns wide",
                  status = "danger", "Status is DANGER!",
                  width = 6, height = 200)
            )
    ),
    tabItem(tabName = "cars",
            h2("What do you want to know about Cars?"),
            plotOutput("myplot"),
            DTOutput("mytable"))
  )
)

```



YOUR TURN

YOUR TURN - WHAT?

- Build a shiny dashboard with at least three tabs
- Minimal requirements:
 - Incorporate at least one (interactive?) plot, three user widgets
 - One of the tabs should include the population numbers published by the census bureau (XXXX.csv)
 - Be creative!

YOUR TURN - HOW & WHEN?

- We will use the break rooms to give teams of about 5 a private room
- Haley, Mattie, and Heike will check in periodically and help solve problems
- Work on the app until 2:15.
- From 2:15 - 3:00 each team will present their app for a few minutes (3-5mins) to all of us