

# Introduction to Dashboards

## BMRN CSM: Building dashboards with R markdown

Martin Frigaard

2020-12-07

# **flexdashboard** = Dashboards using R Markdown (and Shiny)



# Load the packages



```
install.packages(c("tidyverse", "inspectdf",  
                  "flexdashboard", "reactable"))  
  
library(tidyverse)  
library(inspectdf)  
library(flexdashboard)  
library(reactable)
```



# Outline (1)

## Recap rmarkdown

### *What belongs in a dashboard?*

## Layouts

- *Sidebars, Columns, and Rows*
- *Multiple Pages, Tabs*

## Themes

- *Bootstrap themes*



# Outline (2)

## **inspectdf** package

- *graphs, syntax*

## **reactable** package

- *table displays*

## Examples with **shiny**

- *shiny reactivity*



# Materials

## Slides

<https://mjfrigaard.github.io/intro-to-dashboards/Index.html>

## Exercises

## RStudio Project

<https://rstudio.cloud/project/2000287>

**rmarkdown** = **YAML** + **Markdown**  
+ **R** (or other languages)



# *What is RMarkdown?*



## Three technologies:

- 1) Markdown is a plain text markup language for capturing *human-readable* prose
- 2) Data manipulation/graphing/statistical language engines for computing *machine-readable* code
- 3) Multiple *output options* for creating PDFs, Word docs, PowerPoints, HTML, etc.

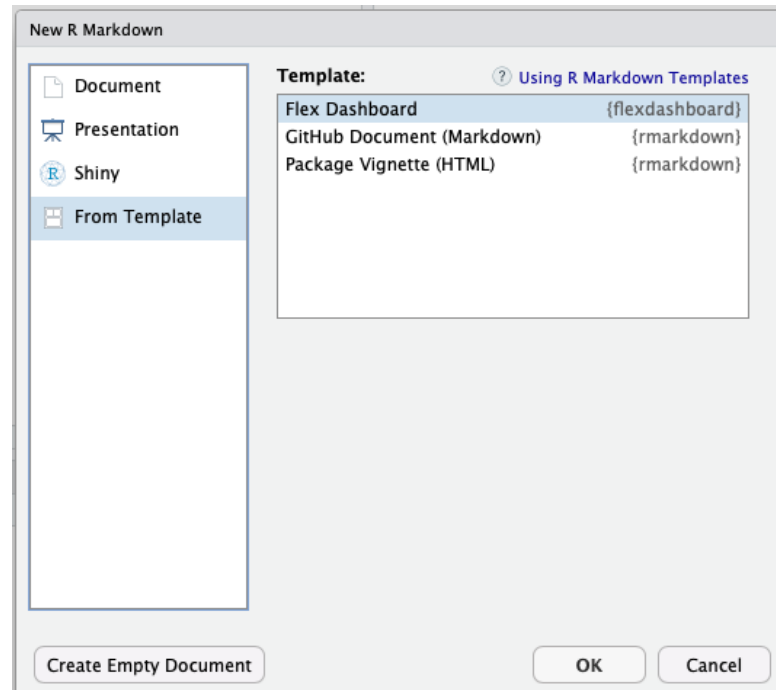


# Your Turn 1



## Open a new R Markdown file

*file > New File > R Markdown > From Template > **flexdashboard***



# Your Turn 2



## Add title and save R Markdown file

The screenshot shows the RStudio interface with an R Markdown file open. The file content is as follows:

```
1 ---
2 title: "My Dashboard"
3 output:
4   flexdashboard::flex_dashboard:
5     orientation: columns
6     vertical_layout: fill
7 ---
8
9 {r setup, include=FALSE}
10 library(flexdashboard)
11
12
13 Column {data-width=650}
14
15
```

An orange box highlights the title line, with an arrow pointing to the text "Add a title".

A "Save File - Untitled1" dialog box is open, showing the file name "my-dashboard.Rmd" in the "File name:" field, which is highlighted with a red box. A red arrow points from this box to the text "Save your file!". The dialog also shows a file explorer view with the following files:

File Name	Size	Modified
..		
.Rhistory	0 B	Dec 6, 2020, 11:27 PM
install.R	175 B	Dec 6, 2020, 11:28 PM
project.Rproj	205 B	Dec 7, 2020, 12:08 AM

The dialog has buttons for "New Folder", "Save", and "Cancel".

# Your Turn 3



knit!

The screenshot shows the RStudio interface with the R Markdown source file on the left and the rendered dashboard on the right. The source file contains a title, output type, flexdashboard options, and three chart sections. The rendered dashboard shows the title, three columns, and three charts. Arrows indicate the mapping from source code to the rendered output.

Source File (my-dashboard.Rmd):

```
1 ---
2 title: "My Dashboard"
3 output:
4   flexdashboard::flex_dashboard:
5     orientation: columns
6     vertical_layout: fill
7 ---
8
9 {r setup, include=FALSE}
10 library(flexdashboard)
11
12
13 Column {data-width=650}
14
15
16 ## Chart A
17 {r}
18
19
20
21
22 Column {data-width=350}
23
24
25 ## Chart B
26 {r}
27
28
29
30
31 ## Chart C
32 {r}
33
34
35
36
37
```

Rendered Dashboard:

- Title: My Dashboard
- Chart A
- Chart B
- Chart C

# What belongs in a dashboard?

Dashboards are particularly common in **business-style reports**. They can be used to **highlight brief and key summaries of a report**. The layout of a dashboard is often grid-based, with components arranged in boxes of various sizes.



# Dashboard Anatomy



The YAML header setting creates the dashboard:

```
output:  
  flexdashboard::flex_dashboard:
```

The layout is determined by the **orientation** and **vertical\_layout** options.

```
orientation: columns  
vertical_layout: fill
```



# Column Widths

Column Widths must add up to **1000**

```
Column {data-width=650}
```

```
### Chart A
```

```
```${r}  
```
```

```
Column {data-width=350}
```

```
### Chart B
```

```
```${r}  
```
```

```
### Chart C
```

```
```${r}  
```
```

# Sidebars



Include a sidebar with `{.sidebar data-width=200}`

```
Inputs {.sidebar data-width=200}
```

```
```\r}\n```\n
```

Adjust the column widths (set both to `{data-width=400}`)

```
Column {data-width=400}
```

```
### Chart A
```

```
```\r}\n```\n
```

```
Column {data-width=400}
```

```
### Chart B
```

```
```\r}\n```\n
```

```
### Chart C
```

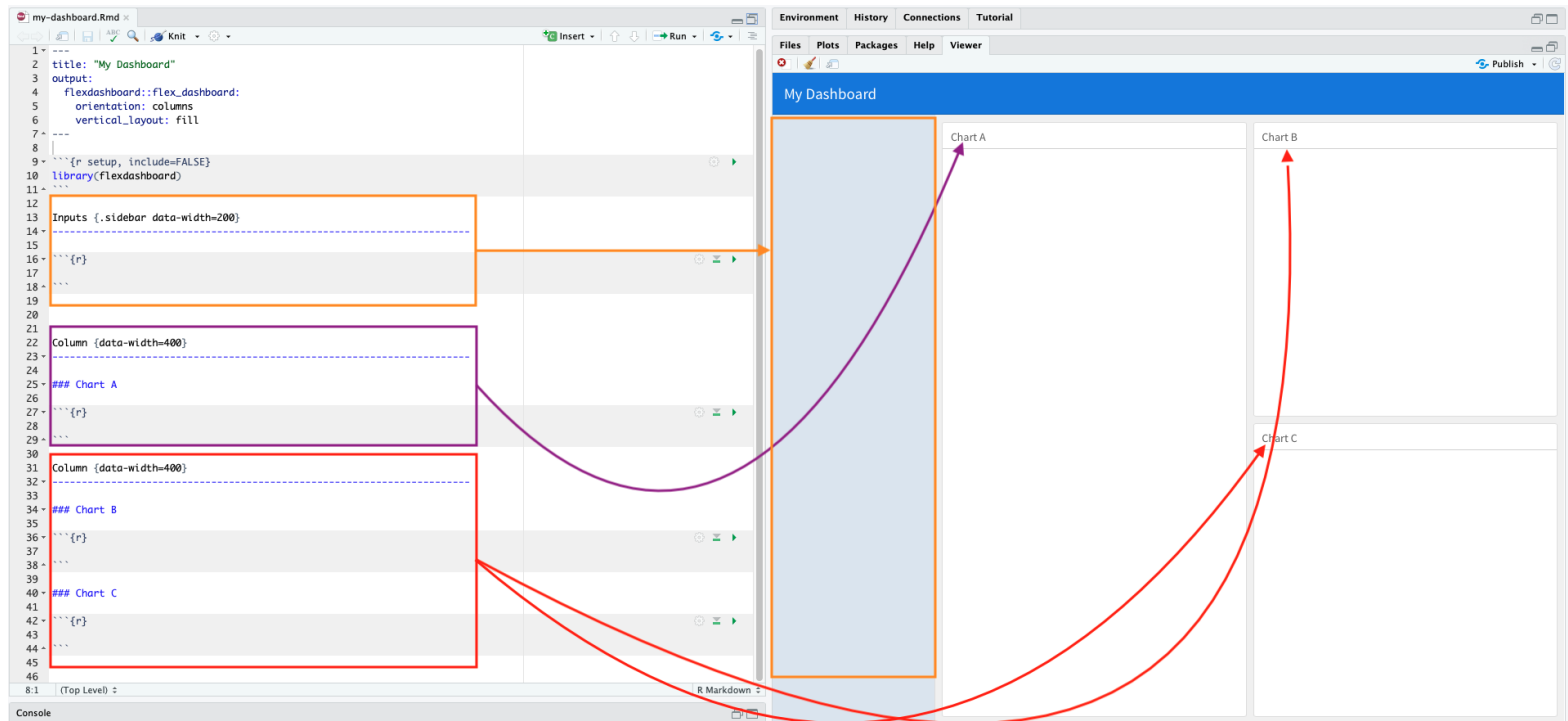
```
```\r}\n```\n
```

## Knit!

# Sidebars



Sidebars are typically used for data inputs and user-interface controls







# Row Layout

## We can also orient by rows

Change the `orientation` of the dashboard

```
output:  
  flexdashboard::flex_dashboard:  
    orientation: columns
```

## Re-knit!

# Rows Layout



The screenshot displays the RStudio interface with an R Markdown file named `my-dashboard.Rmd` open. The code is structured to create a dashboard with two rows. Annotations with colored arrows link specific code blocks to their corresponding visual elements in the dashboard preview:

- Blue arrows:** Point from the `output:` block (lines 3-6) to the top blue header bar and from the `flexdashboard::flex_dashboard:` block (lines 4-6) to the vertical line separating the two rows.
- Purple box:** Encloses the `Inputs` block (lines 13-18), with an arrow pointing to the left sidebar area.
- Red box:** Encloses the first `Column` block (lines 22-29), with an arrow pointing to the area labeled "Row 1".
- Orange box:** Encloses the second `Column` block (lines 31-43), with an arrow pointing to the area labeled "Row 2".

The dashboard preview on the right, titled "My Dashboard", shows the rendered output. It features a blue header, a left sidebar, and two rows of content. "Row 1" contains "Chart A", and "Row 2" contains "Chart B" and "Chart C".

```
1 title: "My Dashboard"
2
3 output:
4   flexdashboard::flex_dashboard:
5     orientation: rows
6     vertical_layout: fill
7
8
9 ---[r setup, include=FALSE]
10 library(flexdashboard)
11
12
13 Inputs {,sidebar data-width=200}
14
15 {r}
16
17
18
19
20
21
22 Column {data-width=400}
23
24 ## Chart A
25
26 {r}
27
28
29
30
31 Column {data-width=400}
32
33 ## Chart B
34
35 {r}
36
37
38
39 ## Chart C
40
41 {r}
42
43
44
45
46
```

# Scrolling



Change the [YAML](#) header back to `orientation: columns` and `vertical_layout: scroll`

```
orientation: columns  
vertical_layout: scroll
```

## Re-knit!

# Scrolling



Now we can scroll past the end of the column.

The screenshot shows the RStudio interface with a file named "my-dashboard.Rmd" open. The code editor on the left contains R Markdown code for a dashboard. A red box highlights the configuration for the first column: `orientation: columns` and `vertical layout: scroll`. A red arrow points from this box to the right-hand pane, which displays the rendered dashboard. The dashboard has a blue header "My Dashboard" and a sidebar on the left. The main content area is divided into two columns. The right column contains a section titled "Chart C". A vertical scrollbar is visible on the right side of the dashboard, indicating that the content can be scrolled vertically. The status bar at the bottom shows "6.28" and "My Dashboard".

# Tabsets



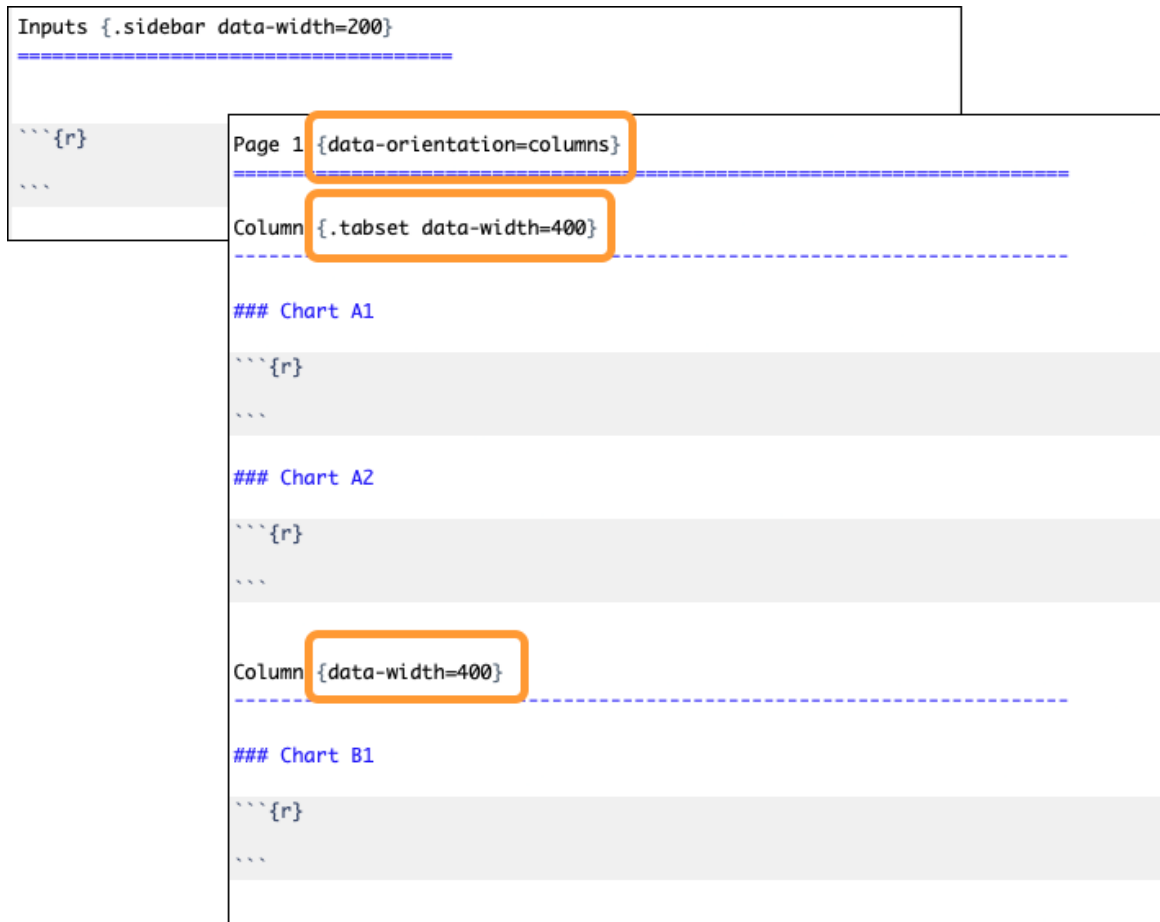
## Add tabsets with `{.tabset}`

The screenshot shows the RStudio interface with a file named 'my-dashboard.Rmd'. The editor on the left contains R Markdown code. A red box highlights a section of the code: a `Column` block with a `.tabset` attribute and three chart placeholders. A red arrow points from this code block to the rendered output on the right. The rendered output, titled 'My Dashboard', shows a tabset with three tabs: 'Chart A1', 'Chart A2', and 'Chart A3'. The 'Chart A2' tab is currently selected and highlighted. Below this tabset, there is another section for 'Chart B' and 'Chart C'. The RStudio interface includes a top menu bar with 'Environment', 'History', 'Connections', and 'Tutorial'. Below the menu is a toolbar with 'Files', 'Plots', 'Packages', 'Help', and 'Viewer'. The 'Viewer' pane on the right shows the rendered dashboard. The console at the bottom shows the output of the R code, indicating that the charts are being rendered.

# Global Sidebar and Pages



For global settings, we use ===== instead of -----



# Global Sidebar and Pages



data-orientation=columns

.tabset

The screenshot displays a dashboard layout generated from R Markdown. On the left is a sidebar containing three chart components, each with a title (Chart A1, Chart A2, Chart B1) and a plot area. The main content area on the right is titled "My Dashboard" and features a tabset with three tabs: "Page 1", "Page 2", and "Page 3". The "Page 1" tab is active, showing a large empty space. The "Page 2" and "Page 3" tabs are also visible but inactive. The layout is organized into columns, with the sidebar on the left and the main content area on the right.

# Global Sidebar and Pages



For global settings, we use ===== instead of -----

```
Page 2 {data-orientation=rows}
=====
Row {,tabset .tabset-fade data-height=600}
-----

### Chart C1

```{r}
```

### Chart C2

```{r}
```

Row {,tabset .tabset-fade data-height=400}
-----

### Chart D1

```{r}
```

### Chart D2

```{r}
```
```



# Global Sidebar and Pages



data-orientation=rows

.tabset-fade

The screenshot displays an R Markdown document on the left and its rendered output on the right. The document on the left is titled 'Page 2 (data-orientation=rows)' and contains a table with four rows. Each row represents a chart, with the first two columns containing chart titles ('Chart C1', 'Chart C2', 'Chart D1', 'Chart D2') and the third column containing a placeholder for the chart content. The rendered output on the right shows a dashboard with a blue header bar containing 'My Dashboard', 'Page 1', and 'Page 2'. The main content area is divided into two sections: the top section contains 'Chart C1' and 'Chart C2', and the bottom section contains 'Chart D1' and 'Chart D2'. The sidebar on the left is a light blue vertical bar.

# Menus



data-navmenu=More

Page 3 {data-navmenu='More'}

Column

### Chart E

```\{r\}

```\

Page 4 {data-navmenu='More'}

Column

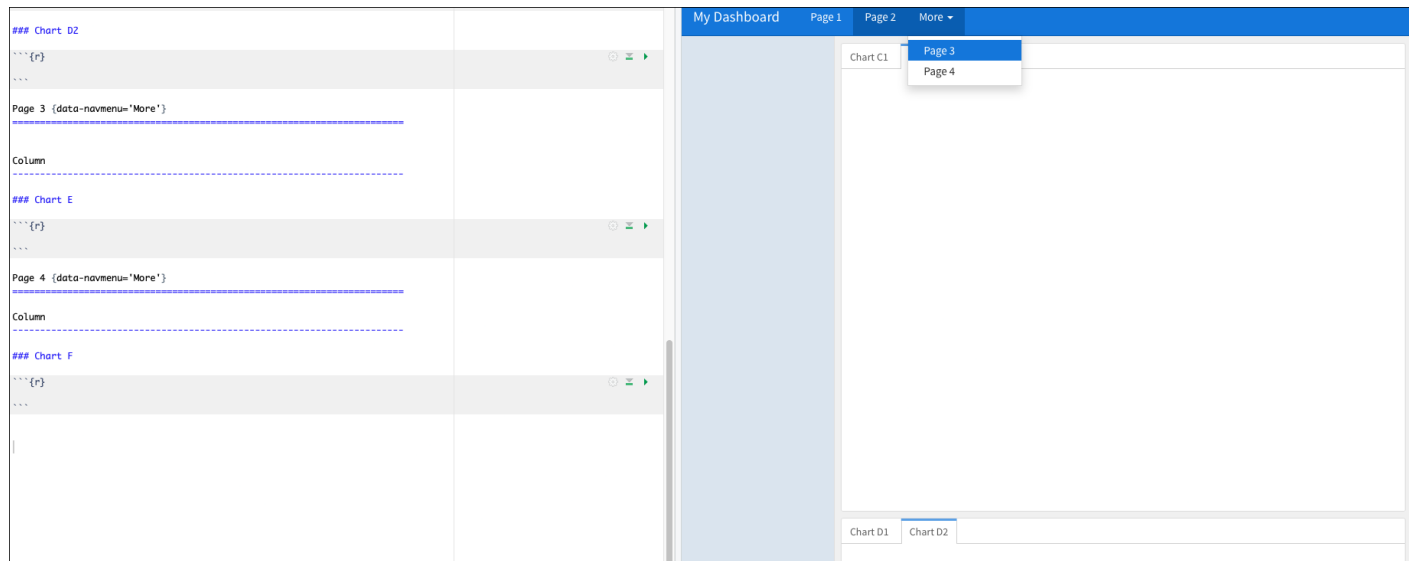
|

### Chart F

```\{r\}

```\

# Menus



# Themes



## Change themes (just like `html_document()`)

```
title: "My Dashboard"
output:
  flexdashboard::flex_dashboard:
    theme: spacelab
```

See the website for more information

**inspectdf** = quickly examine  
datasets



# Previous Slides: Apple Mobility Data



<https://mjfrigaard.github.io/data-viz-as-comm/Index.html>

## Import Data

```
AppleMobRaw <- readr::read_csv("https://bit.ly/36tTVpe")
```

# Previous Slides: Apple Mobility Data



## Don't Forget Wrangling Steps!

```
AppleMobRaw %>%  
  # transpose data  
  tidyr::pivot_longer(cols = -c(geo_type:country),  
    names_to = "date", values_to = "dir_request") %>%  
    # remove missing country data  
  dplyr::filter(!is.na(country) & !is.na(`sub-region`)) %>%  
    # clean names  
  janitor::clean_names() %>%  
    # date  
  mutate(date = lubridate::ymd(date)) %>%  
    # create trans_type  
  rename(trans_type = transportation_type) -> TidyApple
```

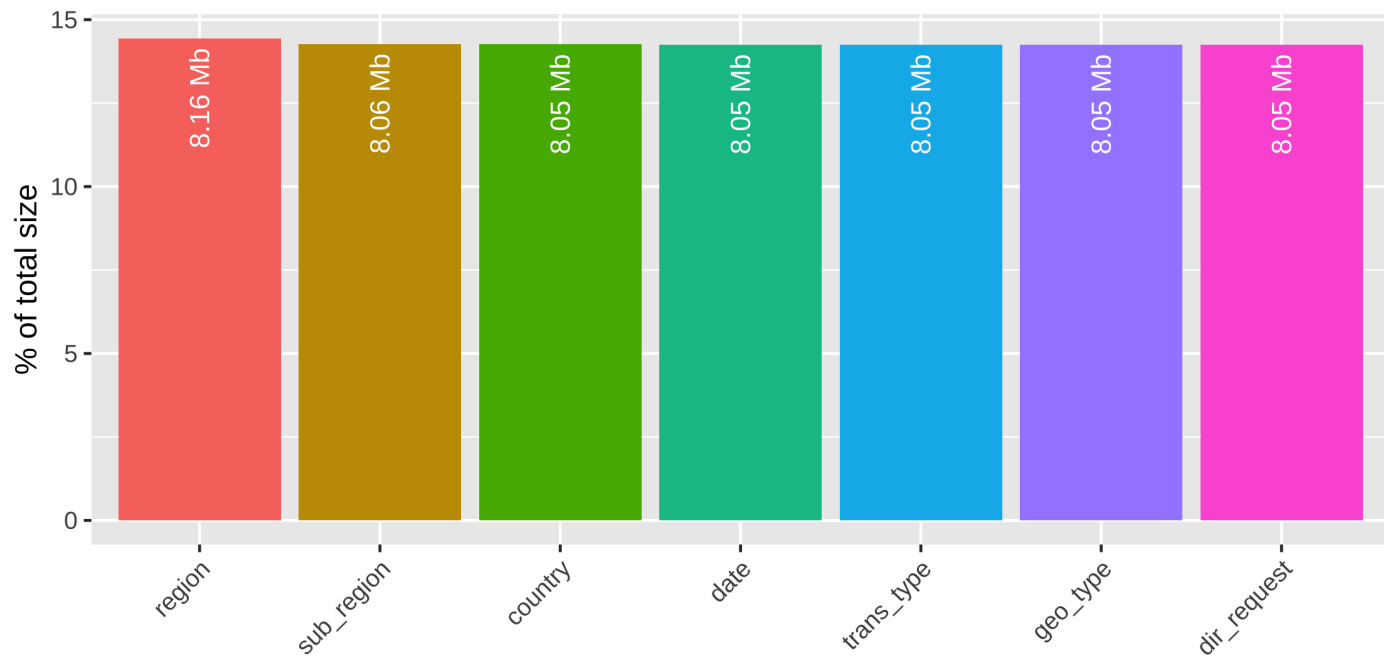
# Dataset size in memory



```
TidyApple %>%  
  inspectdf::inspect_mem() %>%  
  inspectdf::show_plot(text_labels = TRUE)
```

Column sizes in df::.

df::. has 7 columns, 1055293 rows & total size of 56.48 Mb







# Sidebar

## Add the data to the `.sidebar`

Add the `import` and `wrangle` code to the sidebar in the dashboard.

```
# import ----
AppleMobRaw <- readr::read_csv("https://bit.ly/36tTVpe")
# wrangle ----
AppleMobRaw %>%
  # transpose data
  tidyr::pivot_longer(cols = -c(geo_type:country),
    names_to = "date", values_to = "dir_request") %>%
  # remove missing country data
  dplyr::filter(!is.na(country) & !is.na(`sub-region`)) %>%
  # clean names
  janitor::clean_names() %>%
  # date
  mutate(date = lubridate::ymd(date)) %>%
  # create trans_type
  rename(trans_type = transportation_type) -> TidyApple
```

# Page 1, Column 1, Tab 1



## Add the 'Memory Size' Graph

Add this code to [A1](#)

```
TidyApple %>%  
  inspectdf::inspect_mem() %>%  
  inspectdf::show_plot(text_labels = TRUE)
```

## Knit--how does it look?

# Page 1, Column 1, Tab 2



## Add the Missing Data Graph

Add this code to [A2](#)

```
TidyApple %>%  
  inspectdf::inspect_na() %>%  
  inspectdf::show_plot(text_labels = TRUE)
```

## Knit--how does it look?

# Page 1, Column 2, Tab 1



## Add the Categorical Data Graph

Add this code to [B1](#)

```
TidyApple %>%  
  select_if(is.character) %>%  
  inspectdf::inspect_cat() %>%  
  inspectdf::show_plot(text_labels = TRUE)
```

## Knit--how does it look?

# Page 1, Column 2, Tab 2



## Add the **Data Imbalances** Graph

Add this code to [B2](#)

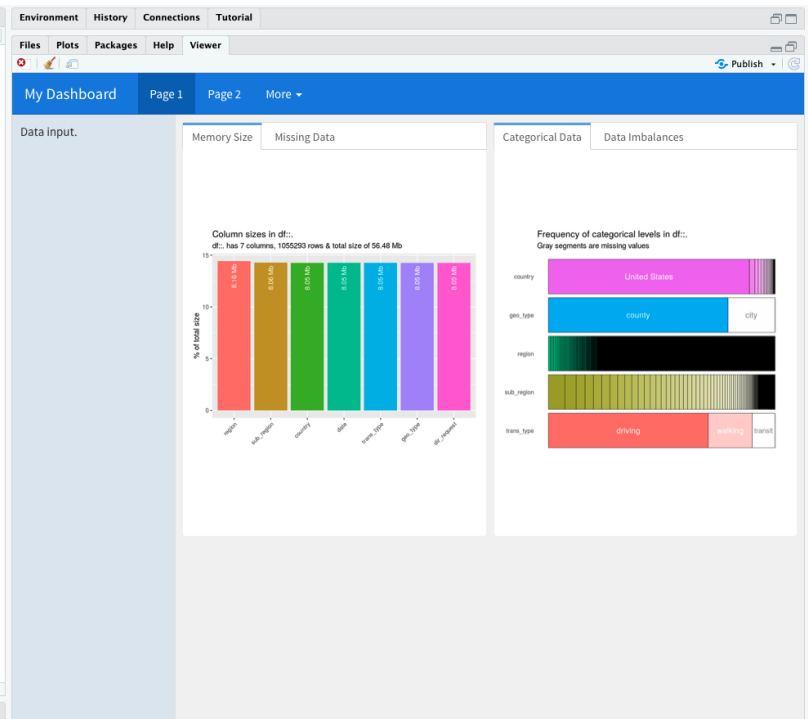
```
TidyApple %>%  
  inspectdf::inspect_imb() %>%  
  inspectdf::show_plot(text_labels = TRUE)
```

## Knit--how does it look?

# Page 1



```
install.R x solutions.Rmd x exercises.Rmd x
Insert Run
38
39
40 Page 1 {data-orientation=columns}
41 -----
42
43 Column {.tabset data-width=400}
44 -----
45
46 ### Memory Size
47
48 ```{r inspect_mem}
49 TidyApple %>%
50   inspectdf::inspect_mem() %>%
51   inspectdf::show_plot(text_labels = TRUE)
52 ```
53
54 ### Missing Data
55
56 ```{r inspect_na}
57 TidyApple %>%
58   inspectdf::inspect_na() %>%
59   inspectdf::show_plot(text_labels = TRUE)
60 ```
61
62
63 Column {.tabset data-width=400}
64 -----
65
66 ### Categorical Data
67
68 ```{r inspect_cat}
69 TidyApple %>%
70   select_if(is.character) %>%
71   inspectdf::inspect_cat() %>%
72   inspectdf::show_plot(text_labels = TRUE)
73 ```
74
75 ### Data Imbalances
76
77 ```{r inspect_imb}
78 TidyApple %>%
79   inspectdf::inspect_imb() %>%
80   inspectdf::show_plot(text_labels = TRUE)
81 ```
82
105.58 [x] Chunk 8: ggribbles [x] R Markdown [x] Console
```



# Page 2 (Rows)



Page 2 {data-orientation=rows}

=====



# Page 2, Row 1, Tab 1

## Create a `.tabset/.tabset-fade` Row

```
Row {.tabset .tabset-fade data-height=600}
```

-----

## Add Numeric Data Graph

```
TidyApple %>%  
  select_if(is.numeric) %>%  
  inspectdf::inspect_num() %>%  
  inspectdf::show_plot(text_labels = TRUE)
```

## Knit--how does it look?



# Page 2, Row 1, Tab 2



## Add 'Distributions ggridges' Graph

```
library(ggribes)
lab_ridges <- labs(x = "Apple directions requests",
                  y = "Transportation Types",
                  title = "Direction Requests by Transportation Type",
                  subtitle = "source: https://covid19.apple.com/mobility")

TidyApple %>%
  ggplot() +
  geom_density_ridges(aes(x = dir_request,
                        y = trans_type,
                        fill = trans_type),
                    alpha = 1/5) +
  lab_ridges
```

## Knit--how does it look?

# Page 2, Row 2, Tab 1



## Create Another `.tabset/.tabset-fade` Row

```
Row {.tabset .tabset-fade data-height=400}
```

-----

## In tab 1, add `TopUSCities` as `paged_table`

```
TopUSCities <- TidyApple %>%  
  filter(country == "United States" &  
         region %in% c("New York City", "Los Angeles",  
                       "Chicago", "Houston", "Phoenix"))  
rmarkdown::paged_table(TopUSCities)
```

## Knit--how does it look?



## Page 2, Row 2, Tab 2

In tab 2, add **MaxUSCitiesDriving** as **reactable**

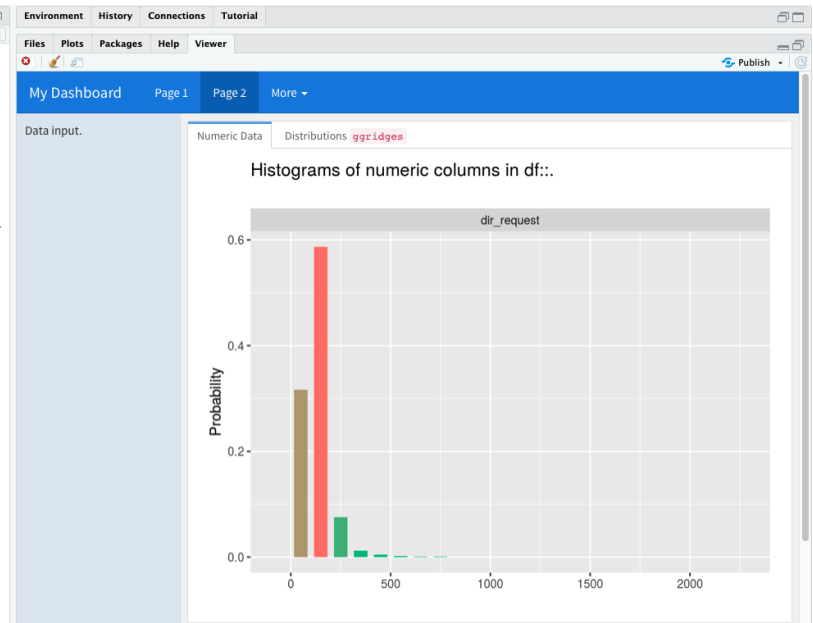
```
TopUSCities %>%  
  filter(trans_type == "driving") %>%  
  group_by(region) %>%  
  slice_max(dir_request) %>%  
  ungroup() -> MaxUSCitiesDriving  
reactable(MaxUSCitiesDriving,  
           resizable = TRUE, showPageSizeOptions = TRUE,  
           selection = "multiple", onClick = "select")
```

**Knit--how does it look?**

# Page 2



```
83 Page 2 {data-orientation=rows}
84 -----
85
86 Row {.tabset, tabset-fade data-height=600}
87 -----
88
89 ### Numeric Data
90
91 ```{r}
92 TidyApple %>%
93   select_if(is.numeric) %>%
94   inspectdf::inspect_num() %>%
95   inspectdf::show_plot(text_labels = TRUE)
96 ```
97
98 ### Distributions 'ggridges'
99
100 ```{r ggridges, message=FALSE, warning=FALSE}
101 library(ggridges)
102 lab_ridges <- labs(x = "Apple directions requests",
103                   y = "Transportation Types",
104                   title = "Direction Requests by Transportation Type",
105                   subtitle = "source: https://covid19.apple.com/mobility")
106
107 TidyApple %>%
108   ggplot() +
109     geom_density_ridges(aes(x = dir_request,
110                             y = trans_type,
111                             fill = trans_type,
112                             alpha = 1/5)) +
113     lab_ridges
114 ```
115
116 Row {.tabset, tabset-fade data-height=400}
117 -----
118
119
120 ### 'TopUSCities' as 'paged_table'
121
122 ```{r TopUSCities}
123 TopUSCities <- TidyApple %>%
```



# Page 2



```
99 - ## Distributions 'ggridges'
100
101 - ```{r ggridges, message=FALSE, warning=FALSE}
102   library(ggridges)
103   lab_ridges <- labs(x = "Apple directions requests",
104                     y = "Transportation Types",
105                     title = "Direction Requests by Transportation Type",
106                     subtitle = "source: https://covid19.apple.com/mobility")
107
108
109   TidyApple %>%
110     ggplot() +
111     geom_density_ridges(aes(x = dir_request,
112                           y = trans_type,
113                           fill = trans_type,
114                           alpha = 1/5) +
115     lab_ridges
116 - ```
117
118   Row {.tabset .tabset-fade data-height=400}
119 -
120
121 - ## 'TopUSCities' as 'paged_table'
122
123 - ```{r TopUSCities}
124   TopUSCities <- TidyApple %>%
125     filter(country == "United States" &
126            region %in% c("New York City", "Los Angeles",
127                          "Chicago", "Houston", "Phoenix"))
128   rmarkdown::paged_table(TopUSCities)
129 - ```
130
131
132 - ## 'MaxUSCitiesDriving' as 'reactable'
133
134 - ```{r MaxUSCitiesDriving}
135   TopUSCities %>%
136     filter(trans_type == "driving") %>%
137     group_by(region) %>%
138     slice_max(dir_request) %>%
139     ungroup() -> MaxUSCitiesDriving
140   reactable(MaxUSCitiesDriving,
141             resizable = TRUE, showPageSizeOptions = TRUE,
142             selection = "multiple", onClick = "select")
143 - ```
144
60.4 Missing Data
```

Environment History Connections Tutorial

Files Plots Packages Help Viewer

My Dashboard Page 1 Page 2 More

Data input.

TopUSCities as paged\_table

MaxUSCitiesDriving as reactable

| <input checked="" type="checkbox"/> | geo_type | region        | trans_type | sub_region | country       | date       | dir_reques |
|-------------------------------------|----------|---------------|------------|------------|---------------|------------|------------|
| <input checked="" type="checkbox"/> | city     | Chicago       | driving    | Illinois   | United States | 2020-07-17 | 166.1      |
| <input checked="" type="checkbox"/> | city     | Houston       | driving    | Texas      | United States | 2020-02-14 | 146.       |
| <input checked="" type="checkbox"/> | city     | Los Angeles   | driving    | California | United States | 2020-02-14 | 152.0      |
| <input checked="" type="checkbox"/> | city     | New York City | driving    | New York   | United States | 2020-09-04 | 152.0      |
| <input checked="" type="checkbox"/> | city     | Phoenix       | driving    | Arizona    | United States | 2020-02-29 | 142.6      |



# More Examples

Check out the package website and gallery

<https://rmarkdown.rstudio.com/flexdashboard/examples.html>