September Event
13.9.16

@RLadiesLondon
#rstats

R - Ladies London

Shiny Apps Knowledge Share: Developing Data Products with R

You MUST have the following installed to participate:

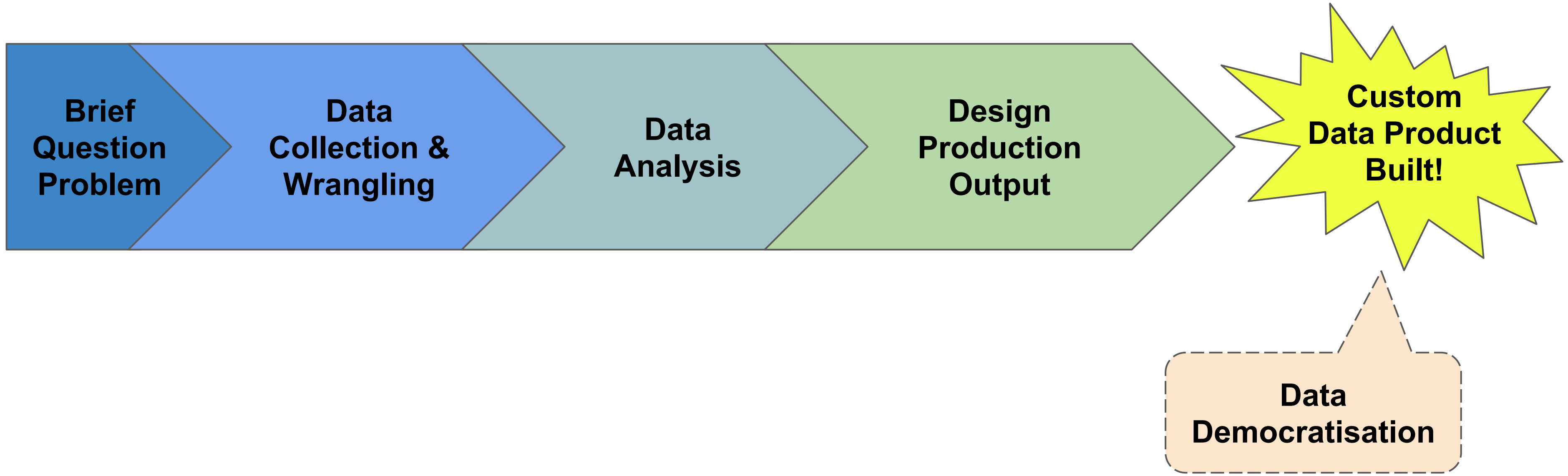
- R & RStudio
- shiny package
- ggplot2 package

Agenda

- Example Use-Case: Diamonds
- ui & server components
- File & Directory instructions
- Try building some apps: 9 Examples
- Try the Example Use-Case
- Replicate Chiin’s Explorer Tool

Teaching you
to FISH!!

Example Use-Case: Diamonds



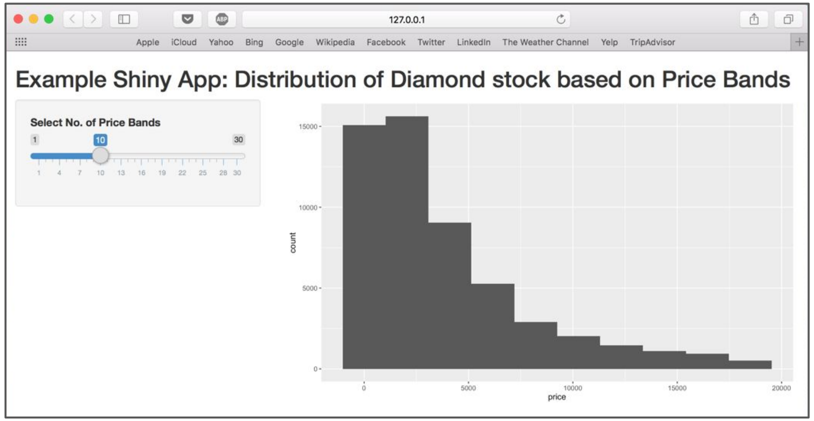
Shiny App: ui & server components

ui object:
Code for creating the web page

server object:
Code for the computer running the R calculations (e.g. your laptop, a server somewhere)

shinyApp(ui = **ui**, server = **server**)

EXAMPLE OUTPUT

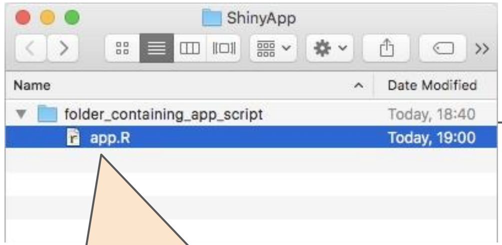

127.0.0.1
Apple iCloud Yahoo Bing Google Wikipedia Facebook Twitter LinkedIn The Weather Channel Yelp TripAdvisor

Example Shiny App: Distribution of Diamond stock based on Price Bands

Select No. of Price Bands

1
10
30
1 4 7 10 13 16 19 22 25 28 30
15000
10000
5000
0
count
0
5000
10000
15000
20000
price

Shiny App: File & Directory instructions



folder_containing_app_script Today, 18:40
app.R Today, 19:00

Making “single-file” shiny apps:

- create one script
- call it “app.R”
- save app.R in it’s own directory by itself
- Note: there is an alternative, original method involving creation of two separate files, ui.R and server.R with a minor difference in code syntax

Template for app.R script

```
#Remember to load & define variables
for the global environment,
e.g. packages, objects

library(shiny)

ui <- basicPage()

server <- function(input, output) {}

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

Try building some apps - 1. Bare minimum, i.e empty app

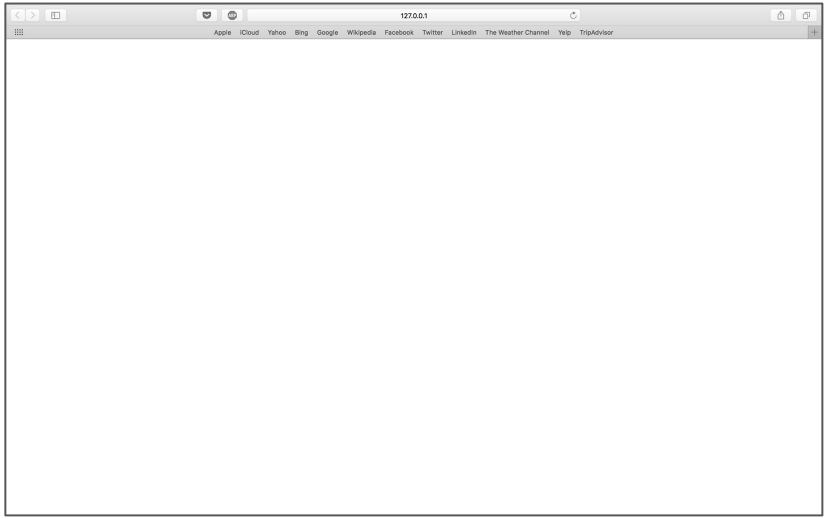
app.R

```
#Remember to load & define variables for the global
environment, e.g. packages, objects

library(shiny)
```

```
ui <- basicPage()
```

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

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



Try building some apps - 2. Basic ui Layout

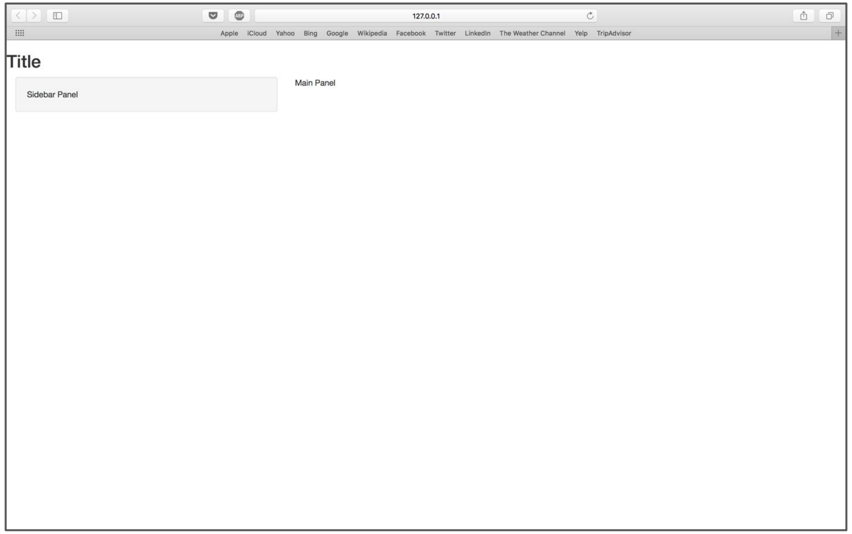
app.R

```
#Remember to load & define variables for the global
environment, e.g. packages, objects

library(shiny)
```

```
ui <- pageWithSidebar(
                        titlePanel(“Title”),
                        sidebarPanel(“Sidebar Panel”),
                        mainPanel(“Main Panel”)
                       )
```

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

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


127.0.0.1
Apple iCloud Yahoo Bing Google Wikipedia Facebook Twitter LinkedIn The Weather Channel Yelp TripAdvisor

Title

Sidebar Panel

Main Panel

Try building some apps - 3. More basic ui Layout

app.R

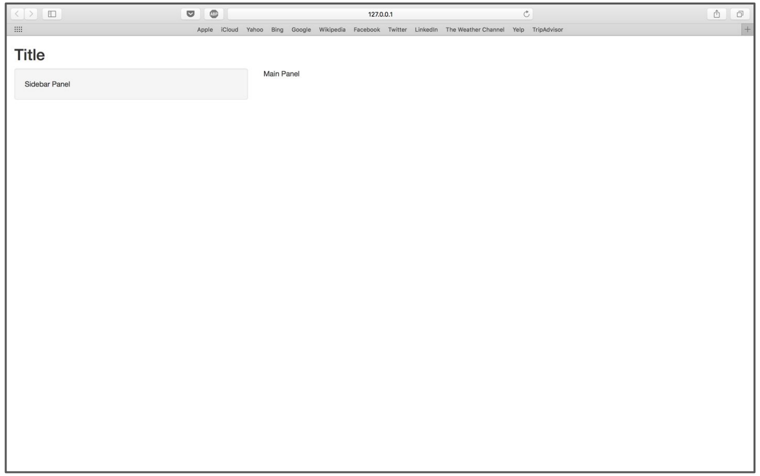
```
#Remember to load & define variables for the global
environment, e.g. packages, objects

library(shiny)
```

```
ui <- fluidPage(
                titlePanel(“Title”),
                sidebarLayout(
                              sidebarPanel(“Sidebar Panel”),
                              mainPanel(“Main Panel”)
                              )
               )
```

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

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

Try resizing your browser and see how the layout changes


127.0.0.1
Apple iCloud Yahoo Bing Google Wikipedia Facebook Twitter LinkedIn The Weather Channel Yelp TripAdvisor

Title

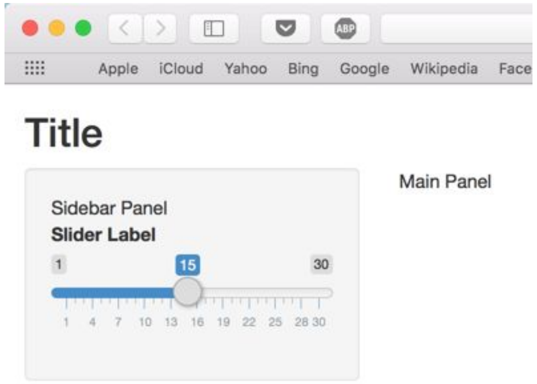
Sidebar Panel

Main Panel

Try building some apps - 4. Basic ui Layout with an Input (slider)

app.R

```
#Remember to load & define variables for the global environment, e.g. packages, objects

library(shiny)

ui <- fluidPage(
                        titlePanel("Title"),
                        sidebarLayout(
                                      sidebarPanel("Sidebar Panel",
                                                    sliderInput(
                                                                 inputId="bins",
                                                                 label="Slider Label",
                                                                 min=1,
                                                                 max=30,
                                                                 value=15
                                                                   )
                                                    ),
                                      mainPanel("Main Panel")
                                      )
                        )

server <- function(input, output) {}

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



Title

Sidebar Panel

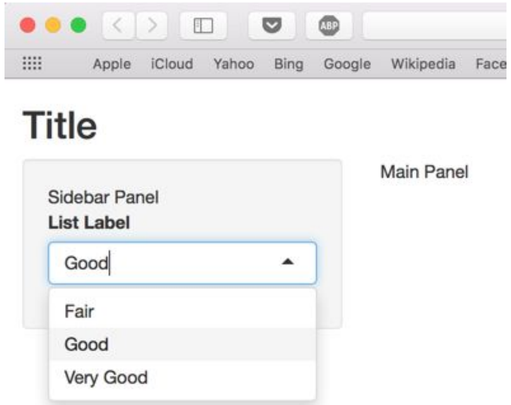
Slider Label

Main Panel

Try building some apps - 5. Basic ui Layout with an Input (list select)

app.R

```
#Remember to load & define variables for the global environment, e.g. packages, objects
library(shiny)

ui <- fluidPage(
            titlePanel(“Title”),
            sidebarLayout(
                        sidebarPanel(“Sidebar Panel”,
                                    selectInput(
                                            inputId=“list”,
                                            label=“List Label”,
                                            choices=c(“Fair”,“Good”,“Very Good”),
                                            selected=“Good”
                                              )
                                    ),
                        mainPanel(“Main Panel”)
             )
        )

server <- function(input, output) {}

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



Title

Sidebar Panel

List Label

Good

Fair

Good

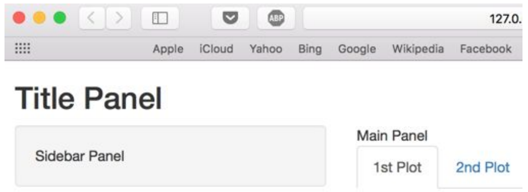
Very Good

Main Panel

Try building some apps - 6. Tab ui Layout with an Output (plot)

app.R

```
#Remember to load & define variables for the global environment, e.g. packages, objects

library(shiny)

ui <- fluidPage(
            titlePanel("Title"),
            sidebarLayout(
                 sidebarPanel("Sidebar Panel"),
                 mainPanel("Main Panel",
                               tabsetPanel(
                                  tabPanel(title="1st Plot", plotOutput(outputId = "plot1")),
                                  tabPanel(title = "2nd Plot", plotOutput(outputId = "plot2"))
                                            )
                               )
                          )
                  )

server <- function(input, output) {}

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



Title Panel

Sidebar Panel

Main Panel

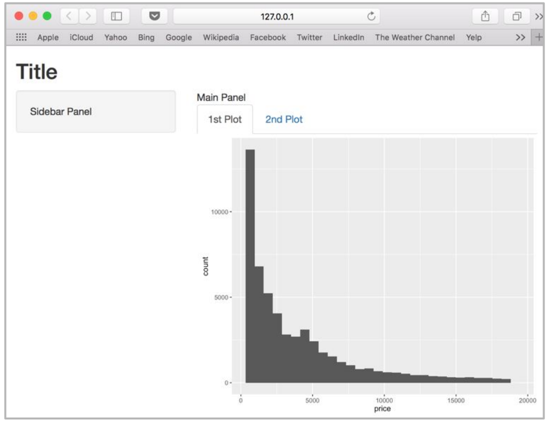
1st Plot

2nd Plot

Try building some apps - 7. Tab ui Layout rendering 1 Output (plot)

app.R

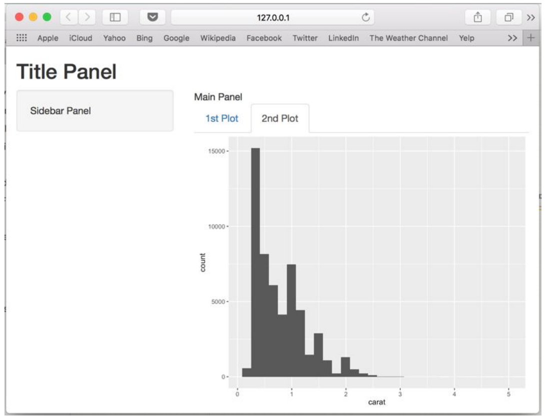
```
#Remember to load & define variables for the global environment, e.g. packages, objects
library(shiny)
library(ggplot2)

ui <- fluidPage(
            titlePanel(“Title”),
            sidebarLayout(
                  sidebarPanel(“Sidebar Panel”),
                  mainPanel(“Main Panel”,
                            tabsetPanel(
                              tabPanel(title=“1st Plot”, plotOutput(outputId = “plot1”)),
                              tabPanel(title = “2nd Plot”, plotOutput(outputId = “plot2”))
                                    )
                          )
                        )
         )

server <- function(input, output) {

                         output$plot1 <- renderPlot({
                          ggplot(data=diamonds, aes(x=price)) + geom_histogram()
                                                      })
                               }

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



Try building some apps - 8. Tab ui Layout rendering 2 Outputs (plot)

app.R

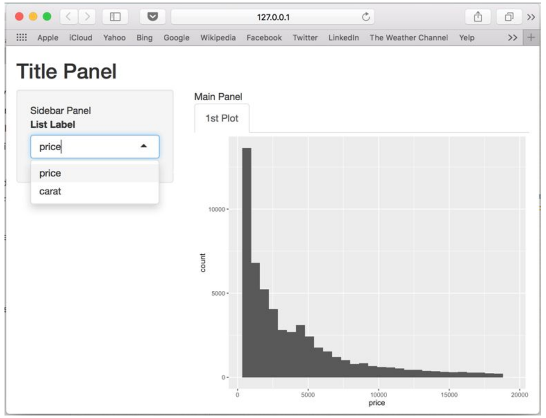
```
#Remember to load & define variables for the global environment, e.g. packages, objects
library(shiny)
library(ggplot2)

ui <- fluidPage(
            titlePanel(“Title”),
            sidebarLayout(
                   sidebarPanel(“Sidebar Panel”),
                   mainPanel(“Main Panel”,
                          tabsetPanel(
                            tabPanel(title=“1st Plot”, plotOutput(outputId = “plot1”)),
                            tabPanel(title = “2nd Plot”, plotOutput(outputId = “plot2”))
                                 )
                          )
                   )
       )

server <- function(input, output) {

                  output$plot1 <- renderPlot({
                     ggplot(data=diamonds, aes(x=price)) + geom_histogram()
                                                  })

                  output$plot2 <- renderPlot({
                     ggplot(data=diamonds, aes(x=carat)) + geom_histogram()
                                                  })
                        }

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


127.0.0.1
Apple iCloud Yahoo Bing Google Wikipedia Facebook Twitter LinkedIn The Weather Channel Yelp
Title Panel
Sidebar Panel
Main Panel
1st Plot
2nd Plot
15000
10000
5000
0
count
0
1
2
3
4
5
carat

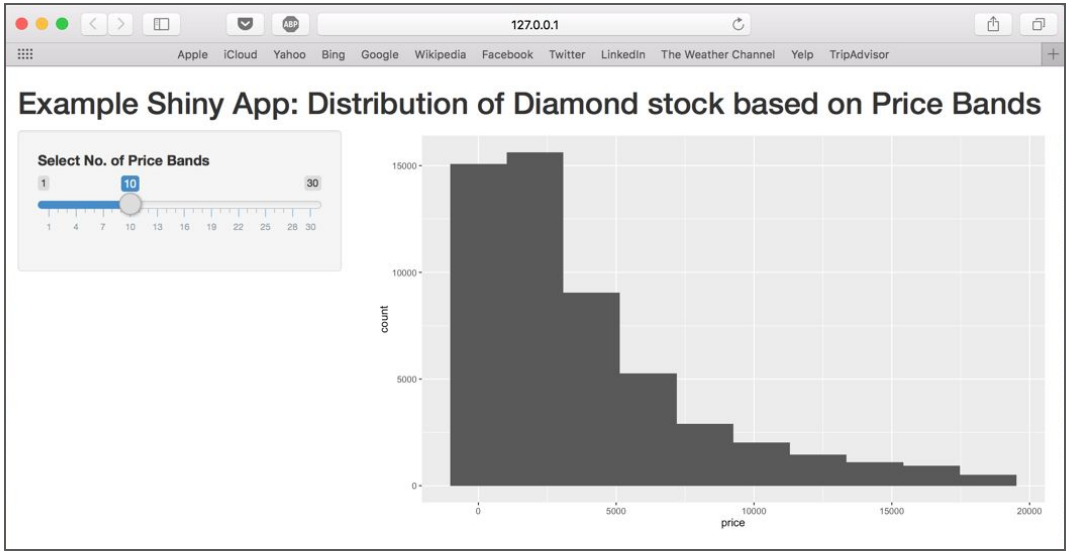
Try building some apps - 9. Reactive ui, single Input-Output (list select & plot)

app.R

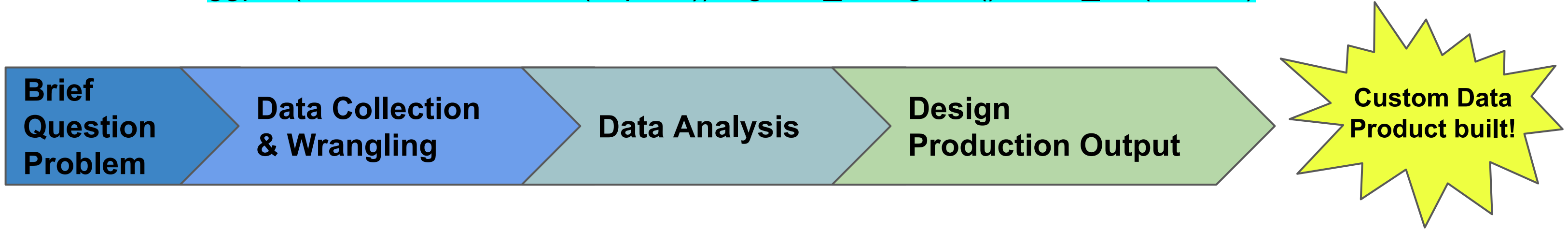
```
#Remember to load & define variables for the global environment, e.g. packages, objects
library(shiny)
library(ggplot2)

ui <- fluidPage(
            titlePanel("Title"),
            sidebarLayout(
                sidebarPanel("Sidebar Panel",
                            selectInput(
                                inputId="list",
                                label="List Label",
                                choices=c("price", "carat"),
                                selected="price"
                                    )
                            ),
                mainPanel("Main Panel",
                            tabsetPanel(
                                tabPanel(title="1st Plot", plotOutput(outputId = "plot1"))
                                    )
                            )
                        )
        )

server <- function(input, output) {

                        output$plot1 <- renderPlot({
                            ggplot(data=diamonds, aes_string(x=input$list)) + geom_histogram()
                                                })
                    }
shinyApp(ui = ui, server = server)
```



Now Try the Example Use-Case


127.0.0.1
Apple iCloud Yahoo Bing Google Wikipedia Facebook Twitter LinkedIn The Weather Channel Yelp TripAdvisor
Example Shiny App: Distribution of Diamond stock based on Price Bands
Select No. of Price Bands
1
10
30
count
price

ggplot(data=diamonds, aes(x=price)) + geom_histogram() + stat_bin(bins=**?**)



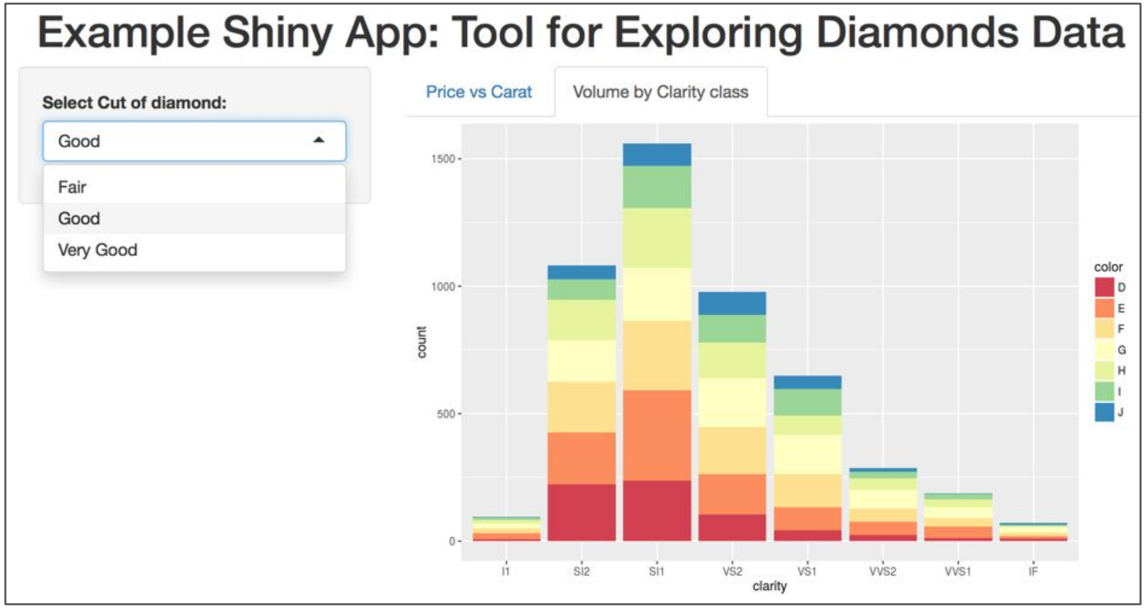
Now Try the Example Use-Case: CODE

```
library(shiny)
library(ggplot2)

ui <- fluidPage(
            titlePanel("Example Shiny App: Distribution of Diamond stock by Price Bands"),
            sidebarLayout(
                        sidebarPanel(
                        sliderInput(inputId="bands",
                                 label="Select No. of Price Bands",
                                 min=1,
                                 max=30,
                                 value=15)
                       ),
            mainPanel(
                     plotOutput(outputId="plot")
                     )
                       )
           )

server <- function(input, output){

                        output$plot <- renderPlot({
                                          ggplot(data=diamonds, aes(x=price)) + geom_histogram() + stat_bin(bins=input$bands)
                                          })
                        }

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

Replicate Chiin’s Explorer Tool

You can deploy a limited no./usage of your apps for free on shinyapps.io (like I’ve done here to demonstrate), which is a freemium hosted version of Shiny Server

chiin.shinyapps.io/diamonds/



Example Shiny App: Tool for Exploring Diamonds Data

Select Cut of diamond: Good

Replicate Chiin’s Explorer Tool: CODE

```
library(shiny)
library(ggplot2)
library(RColorBrewer)

ui <- fluidPage(
                titlePanel("Example Shiny App: Tool for Exploring Diamond Data"),
                sidebarLayout(
                              sidebarPanel(
                                           selectInput(inputId = "cut",
                                                       label = "Select Cut of Diamond:",
                                                       choices = c("Fair", "Good", "Very Good"),
                                                       selected = "Good")
                                                        ),
                              mainPanel(
                                        tabsetPanel(
                                                    tabPanel("Price vs Carat",
                                                             plotOutput(outputId = "scatter")
                                                            ),
                                        tabPanel("Volume by Clarity class",
                                                 plotOutput(outputId = "bar")
                                                  )
                                                    )
                                        )
                              )
                )

server <- function(input, output) {

      getDataset <- reactive({
                   if (input$cut=="Fair") {
                   return(diamonds[diamonds$cut=="Fair", ])
                   } else if (input$cut=="Good") {
                   return(diamonds[diamonds$cut=="Good", ])
                   } else {
                   return(diamonds[diamonds$cut=="Very Good", ])
                   }
                         })

      output$scatter <- renderPlot({
                                ggplot(data=getDataset(), aes(x=price, y=carat)) + geom_point(aes(colour=color)) + scale_color_brewer(palette="Blues")
                               })

      output$bar <- renderPlot({
                              ggplot(data=getDataset(), aes(x=clarity)) + geom_bar(aes(fill=color)) + scale_fill_brewer(palette="Spectral")
                             })
                            }

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

September Event
13.9.16

R- Ladies London

Thanks for coming!

Remember there's more than one way to skin a cat!! (i.e. as with all things in R, there's multiple code to achieve the same outcome, so go with whatever code works best for you!

Recommended resources

-Official site for gallery, tutorials & articles: shiny.rstudio.com

-From UBC STAT 545 course: deanattali.com/blog/building-shiny-apps-tutorial & associated slides

-Blog:zevross.com/blog/2016/04/19/r-powered-web-applications-with-shiny-a-tutorial-and-cheat-sheet-with-40-example-apps

-For specific questions: Stack Overflow!

For access to the slides & scripts email **rladieslondon@gmail.com**
with your request & **your full name**!
(which you'll already have done if you're reading this now!)