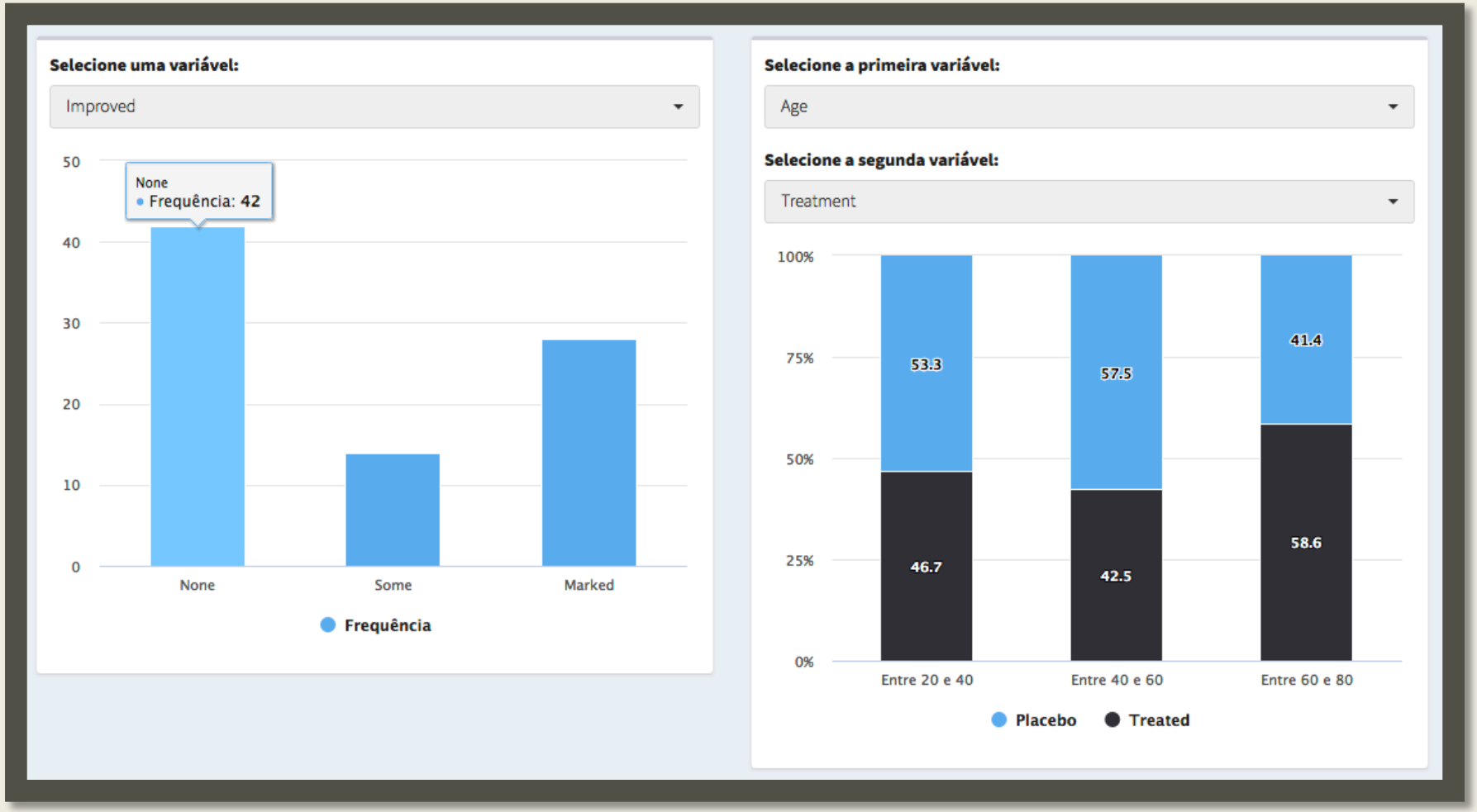
Guia Dashboard
(Shiny — RStudio)

Objetivo

- Por meio desse guia, pretende-se que, ao final, você possa montar e publicar uma dashboard simples, na linguagem de programação R, através dos packages: “**shiny**”, “**shinydashboard**”, e “**shinyWidgets**”.
- Os dados escolhidos como exemplo são do dataset “Arthritis”, que está incluído no package “**vcd**”. Já para a filtragem desses dados, será utilizado o package “**dplyr**”.
- Para montar os gráficos, os exemplos serão feitos através do package “**highcharter**”.
- Por fim, será feito um deploy do aplicativo na nuvem do shinyapps.io.

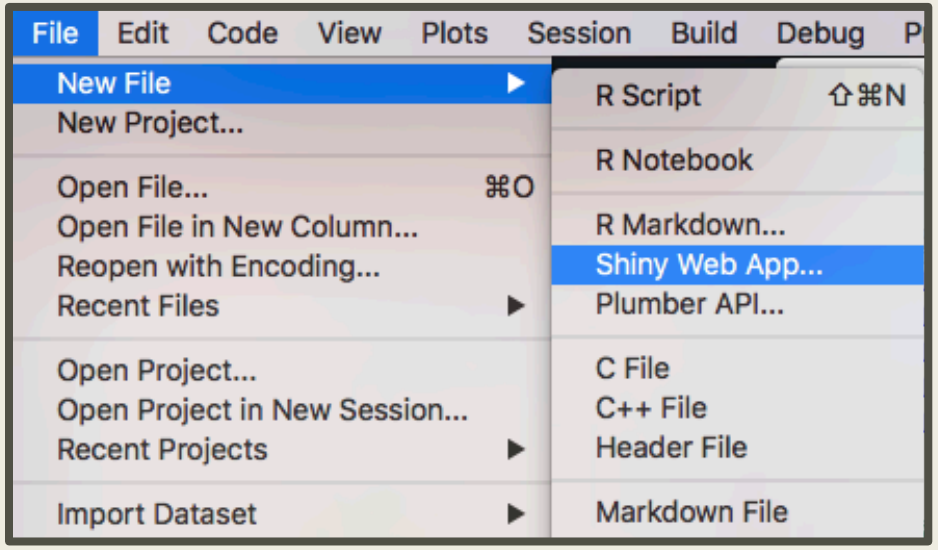
Caso queira criar dashboards mais completas, recomenda-se que acesse os links acima.

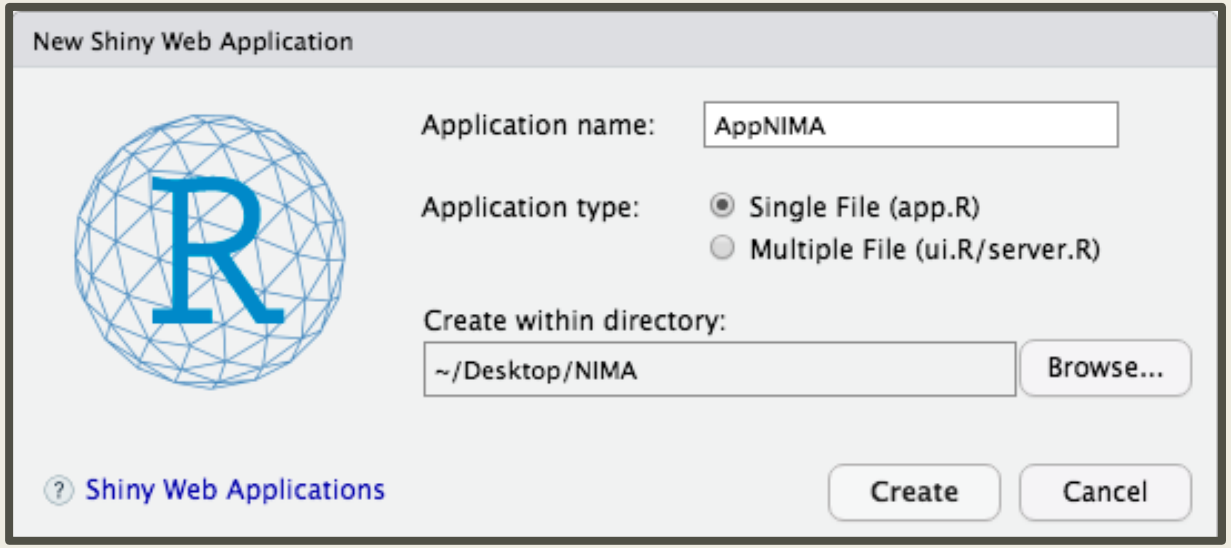
Dashboard pronta: GitHub



Criando o arquivo

- Primeiro, crie um arquivo “**Shiny Web App**”.
- Pode ser arquivo único, de preferência, até que se acostume.





Global

- Primeiramente, vamos colocar as **libraries** que serão mais utilizadas:

```
library(shiny)
library(shinydashboard)
library(shinyWidgets)
```

```
library(highcharter)
library(dplyr)
```

- Depois, filtrar a base de **dados**:

```
# Usar o dataset "Arthritis" que está incluído no package "vcd" para criar um data frame.
df <- data.frame(vcd::Arthritis) %>%
   mutate(Age = cut(Age,
                    breaks = c(20, 40, 60, 80),
                    right = FALSE,
                    labels = c("Entre 20 e 40", "Entre 40 e 60", "Entre 60 e 80")))

# Criar um vetor com os nomes das colunas da nossa base para usar no pickerInput/seletor
variaveis <- setdiff(names(df), "ID")
```

UI

- Indo para a dashboard em si, a sua estrutura é composta pela **interface do usuário** (UI), que é a parte mais visual e onde o usuário vai colocar os seus “inputs”, e pelo **server**, que define como o aplicativo vai funcionar e onde os “inputs” serão transformados em “outputs”.

- A estrutura da UI é composta por:
  - um **Header** (cabeçalho),
  - uma **Sidebar** (barra lateral, usada como menu)
  - e o **Body** (corpo da dashboard).
- Mas, para simplificar, vamos explorar somente o Body.

```
ui <- dashboardPage(
 dashboardHeader(disable = TRUE),
 dashboardSidebar(disable = TRUE),
 dashboardBody()
)

server <- function(input, output) { }

shinyApp(ui, server)
```

UI

- O Body, será composto por duas **box** (caixa), contendo o **pickerInput** (seletor) e o **highchartOutput** (gráfico).
- No caso do seletor, vamos usar apenas três argumentos: inputId (nome de identificação, que deve ser único para cada um), label (texto com rótulo/título) e choices (lista de valor/nomes exibidos como opções de escolha).
- Para o gráfico, basta atribuir um código de identificação, também único para cada um.

```
dashboardBody(
    fluidRow(

      box(pickerInput(inputId = "seletor_univariado",
                      label = "Selecione uma variável:",
                      choices = variaveis),

        highchartOutput(outputId = "grafico_univariado")),


      box(pickerInput(inputId = "seletor_var1",
                      label = "Selecione a primeira variável:",
                      choices = variaveis),

        pickerInput(inputId = "seletor_var2",
                    label = "Selecione a segunda variável:",
                    choices = variaveis),

        highchartOutput(outputId = "grafico_bivariado"))

      )
    )
```

Server → Gráfico univariado

- Primeiro, é preciso montar uma data frame com a tabela a ser utilizada no gráfico. É possível fazer o **filtro** e a **tabela** a parte, para ver como ela vai ficar e depois traduzir para o server.

```
# Selecionar a coluna com alguma das “variaveis”
filtro <- df %>%
   select(variaveis[1])

# Criar uma df, a partir da tabela com a coluna selecionada
tabela <- as.data.frame(table(filtro)) %>%
   setNames(c("variavel", "frequencia"))
```

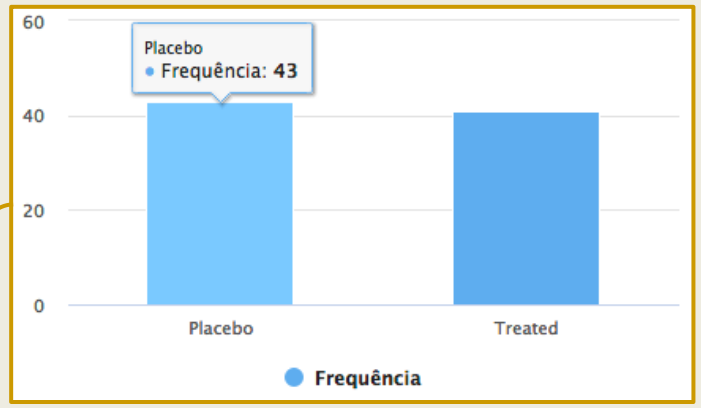
	variavel	frequencia
1	Placebo	43
2	Treated	41

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

   filtro_univariado <- reactive({
      df %>%
         select(input$seletor_univariado)
   })

   data_grafico_univariado <- reactive({
      as.data.frame(table(filtro_univariado())) %>%
         setNames(c("variavel", "frequencia"))
   })

}
```

Server → Gráfico univariado

- A mesma dica também serve para o **gráfico**.

```
highchart() %>%
  hc_xAxis(list(categories = tabela$variavel)) %>%
  hc_add_series(data = tabela,
                name = "Frequência",
                type = "column",
                hcaes(x = variavel, y = frequencia))
```



o gráfico é do tipo coluna, em que os valores do x são os nomes da variavel e do y são os valores de frequencia

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

   output$grafico_univariado <- renderHighchart({
      highchart() %>%
        hc_xAxis(list(categories = data_grafico_univariado()$variavel)) %>%
        hc_add_series(
                data = data_grafico_univariado(),
                name = "Frequência",
                type = "column",
                hcaes(x = variavel, y = frequencia))
   })

}
```

nome das categorias no eixo x é a coluna “variavel” da tabela criada

cria uma série com nome “frequência”, a partir da tabela criada

Server → Gráfico bivariado

```
# Selecionar as duas colunas a serem cruzadas
filtro <- df %>%
   select(variaveis[1], variaveis[2])

# Criar uma df, com a frequência e a porcentagem
tabela <- as.data.frame(table(filtro)) %>%
   setNames(c("Var1", "Var2", "Freq")) %>%
   group_by(Var1) %>%
   mutate(Porcentagem = round(Freq/sum(Freq)*100,
                              digits = 1))
```

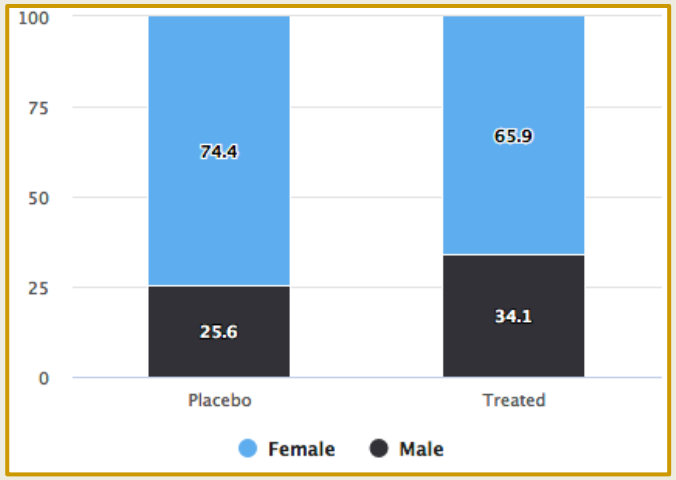
	Var1	Var2	Freq	Porcentagem
1	Placebo	Female	32	74.4
2	Treated	Female	27	65.9
3	Placebo	Male	11	25.6
4	Treated	Male	14	34.1

Para visualizar como uma tabela cruzada,
utilize o as.data.frame.matrix(table(filtro))

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

 filtro_bivariado <- reactive({
      df %>%
         select(input$seletor_var1, input$seletor_var2)
   })

   data_grafico_bivariado <- reactive({
      as.data.frame(table(filtro_bivariado())) %>%
         setNames(c("Var1", "Var2", "Freq")) %>%
         group_by(Var1) %>%
         mutate(Porcentagem = round(Freq/sum(Freq)*100,
                                    digits = 1))
   })

}
```

Server → Gráfico bivariado

```
highchart() %>%
  hc_plotOptions(column = list(stacking = "percent",
                               dataLabels = list(enabled = TRUE))) %>%
  hc_xAxis(list(categories = tabela$Var1)) %>%
  hc_add_series(tabela,
                "column",
                hcaes(x = Var1, y = Porcentagem, group = Var2))
```




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

 output$grafico_bivariado <- renderHighchart({
     highchart() %>%
       hc_plotOptions(column = list(stacking = "percent",
                                    dataLabels = list(enabled = TRUE))) %>%
       hc_xAxis(list(categories = data_grafico_bivariado()$Var1)) %>%
       hc_add_series(data_grafico_bivariado(),
                     "column",
                     hcaes(x = Var1, y = Porcentagem, group = Var2))
   })

}
```

coluna agrupada, com rótulo de dados

Publicação

- Primeiro, é preciso criar uma conta no shinyapps.io.
- Depois, basta logar essa conta no RStudio, seguindo as instruções que aparecem, e publicar.