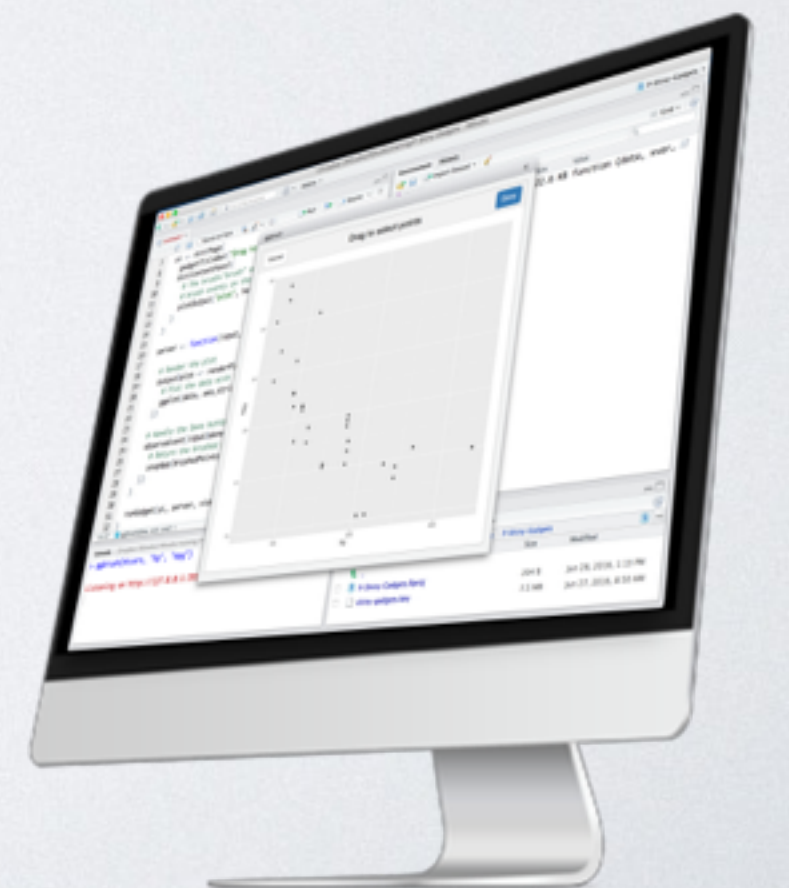


# SHINY GADGETS

INTERACTIVE TOOLS FOR PROGRAMMING



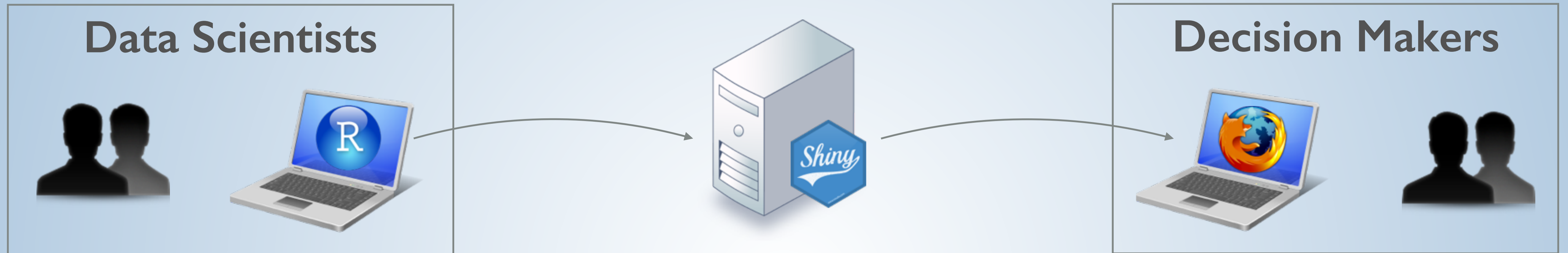
# WHAT IS A GADGET?





# *Shiny app*

Shiny Server



# Shiny gadget

Shiny Server



**DEMO**



# Shiny gadget



Why?

**Programming** ..... **Communicating**

Where?

**In local session** ..... **On Network**

How?

**Invoked** ..... **Deployed**

Who?

**Data Scientist** ..... **Decision Maker**



"Not only am I a Shiny developer,  
I am also a Shiny client."



# *Potential?*

Data cleaner

Model/simulation UI

Plot builder

Data filterer

Leverage analysis

Color picker

Data store UI

Things that are hard to do at the command line,  
but are part of a reproducible workflow.

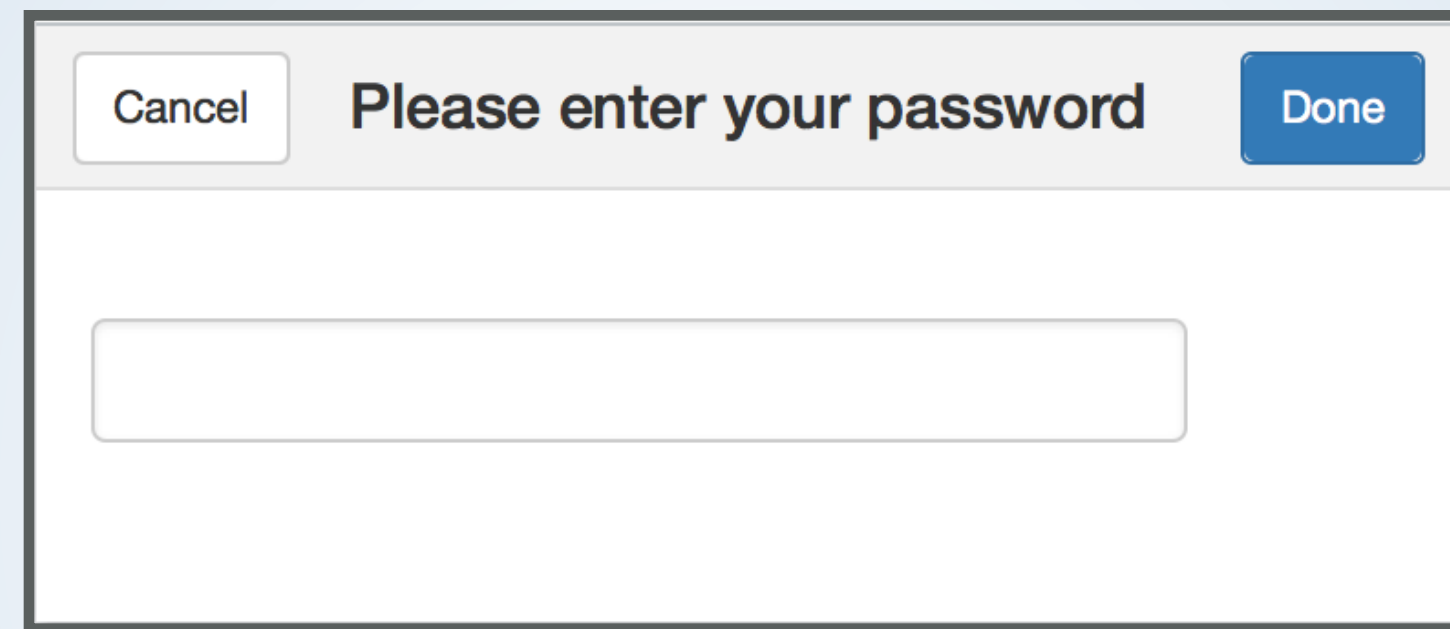
*get\_password.R*

A dialog box with a light gray header bar. The header bar contains three elements: a 'Cancel' button on the left, the text 'Please enter your password' in the center, and a 'Done' button on the right. The 'Cancel' button is white with a gray border, and the 'Done' button is blue with white text. Below the header bar is a large white area containing a single text input field with a light gray border.



# *3 Essential Features*

2. `library(miniUI)`



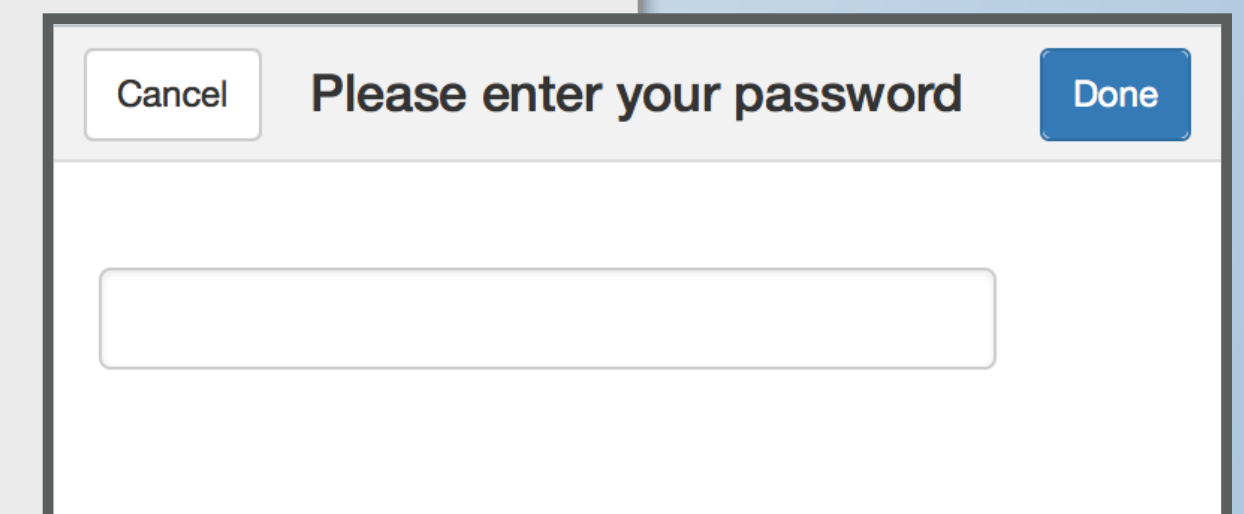
A screenshot of a password prompt dialog box. The dialog has a title bar with the text "Please enter your password". Below the title bar, there is a text input field. To the left of the input field is a "Cancel" button, and to the right is a "Done" button. The dialog is centered on the screen.

1. `get_password()`

3. `stopApp()`

Programming Workflow

```
get_password <- function() {  
  ui <- miniPage(  
    gadgetTitleBar("Please enter your password"),  
    miniContentPanel(  
      passwordInput("password", "")  
    )  
  )  
  
  server <- function(input, output) {  
    observeEvent(input$done, {  
      stopApp(input$password)  
    })  
    observeEvent(input$cancel, {  
      stopApp(stop("No password.", call. = FALSE))  
    })  
  }  
  
  runGadget(ui, server,  
    viewer = dialogViewer("Password"))  
}  
get_password()
```



A screenshot of a Shiny dialog box. The title bar contains the text "Please enter your password" and two buttons: "Cancel" on the left and "Done" on the right. Below the title bar is a single-line text input field, currently empty.



```
get_password <- function() {  
  ui <- miniPage(  
    gadgetTitleBar("Please enter your password"),  
    miniContentPanel(  
      passwordInput("password", "")  
    )  
  )  
  
  server <- function(input, output) {  
    observeEvent(input$done, {  
      stopApp(input$password)  
    })  
    observeEvent(input$cancel, {  
      stopApp(stop("No password.", call. = FALSE))  
    })  
  }  
  
  runGadget(ui, server,  
    viewer = dialogViewer("Password"))  
}  
get_password()
```

1

**Define the app within  
a function.**

**Invoke with function**

```
get_password <- function() {  
  ui <- miniPage(  
    gadgetTitleBar("Please enter your password"),  
    miniContentPanel(  
      passwordInput("password", "")  
    )  
  )  
  
  server <- function(input, output) {  
    observeEvent(input$done, {  
      stopApp(input$password)  
    })  
    observeEvent(input$cancel, {  
      stopApp(stop("No password.", call. = FALSE))  
    })  
  }  
  
  runGadget(ui, server,  
    viewer = dialogViewer("Password"))  
}  
get_password()
```

**2** Use miniUI for  
concise layout

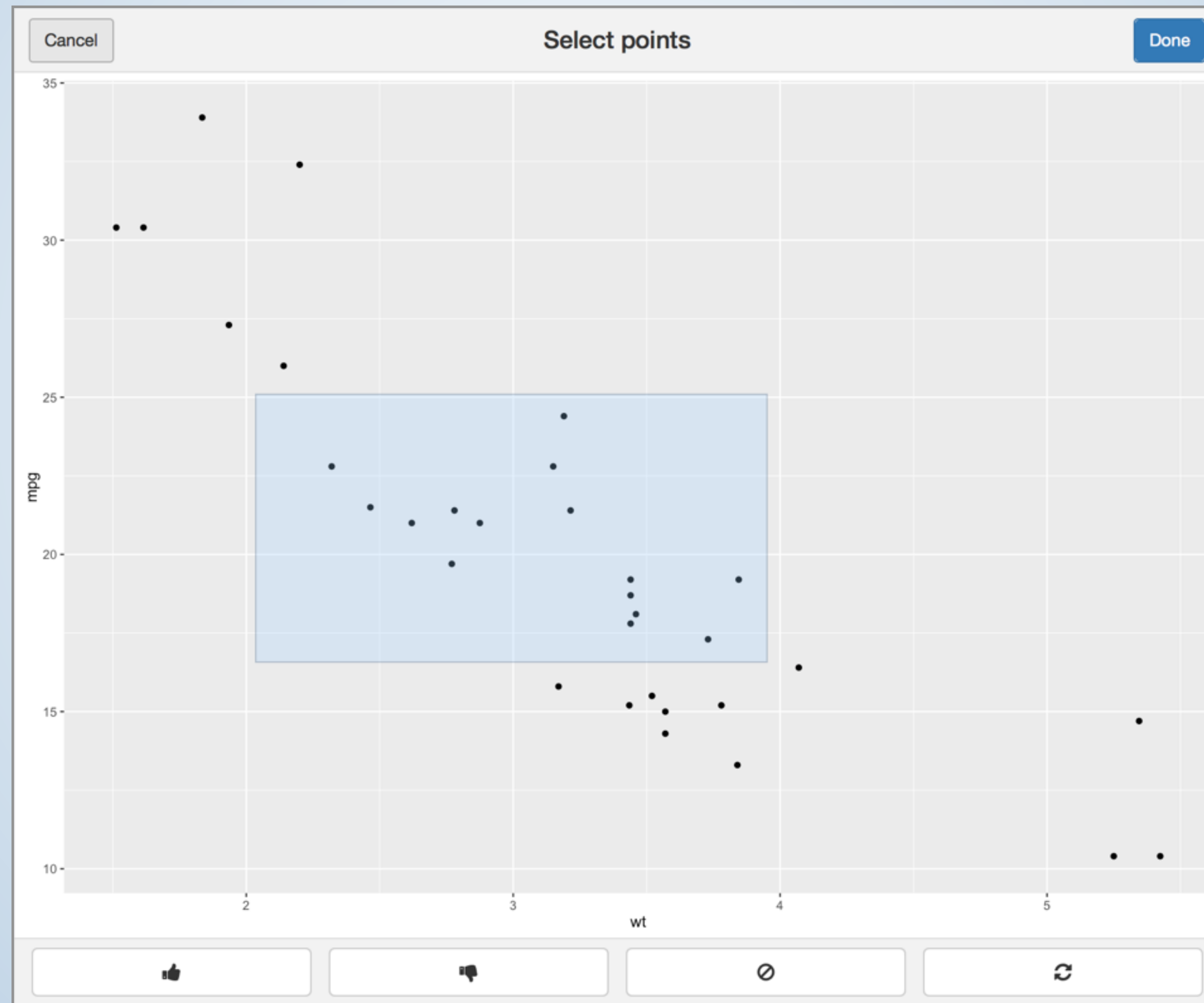


```
get_password <- function() {  
  ui <- miniPage(  
    gadgetTitleBar("Please enter your password"),  
    miniContentPanel(  
      passwordInput("password", "")  
    )  
  )  
  
  server <- function(input, output) {  
    observeEvent(input$done, {  
      stopApp(input$password)  
    })  
    observeEvent(input$cancel, {  
      stopApp(stop("No password.", call. = FALSE))  
    })  
  }  
  
  runGadget(ui, server,  
    viewer = dialogViewer("Password"))  
}  
get_password()
```

3

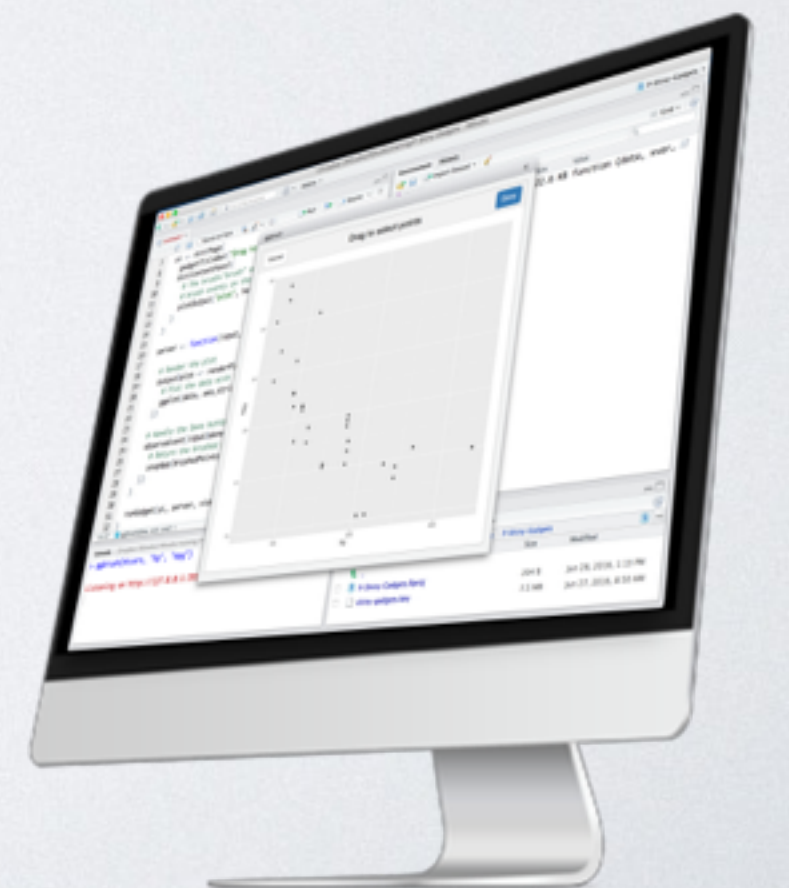
Close app, return  
values with stopApp()

# *brush\_gadget.R*





# I. <- FUNCTION





```

pick_points <- function(data, x, y) {
  ...
  server <- function(input, output) {

    vals <- reactiveValues(keep = rep(TRUE, nrow(data)))

    output$plot1 <- renderPlot({
      keep    <- data[ vals$keep, , drop = FALSE]
      exclude <- data[!vals$keep, , drop = FALSE]

      ggplot(keep, aes_(x, y)) +
        geom_point(data = exclude, color = "grey80") +
        geom_point()

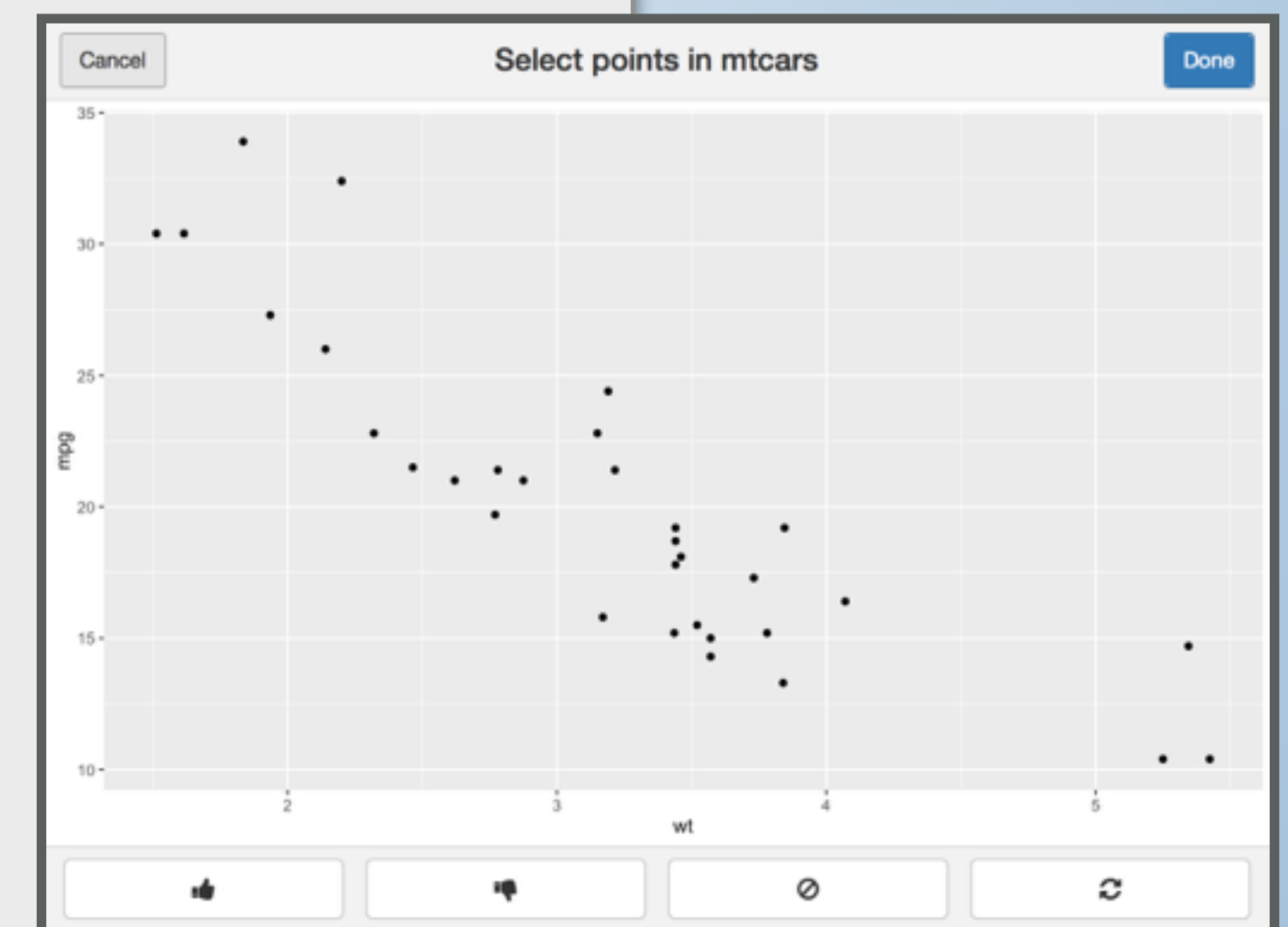
    })

    ...
  }

  runGadget(ui, server)
}

pick_points(mtcars, ~wt, ~mpg)

```





```

pick_points <- function(data, x, y) {
  ...
  server <- function(input, output) {

    vals <- reactiveValues(keep = rep(TRUE, nrow(data)))

    output$plot1 <- renderPlot({
      keep      <- data[ vals$keep, , drop = FALSE]
      exclude <- data[!vals$keep, , drop = FALSE]

      ggplot(keep, aes_(x, y)) +
        geom_point(data = exclude, color = "grey80") +
        geom_point()

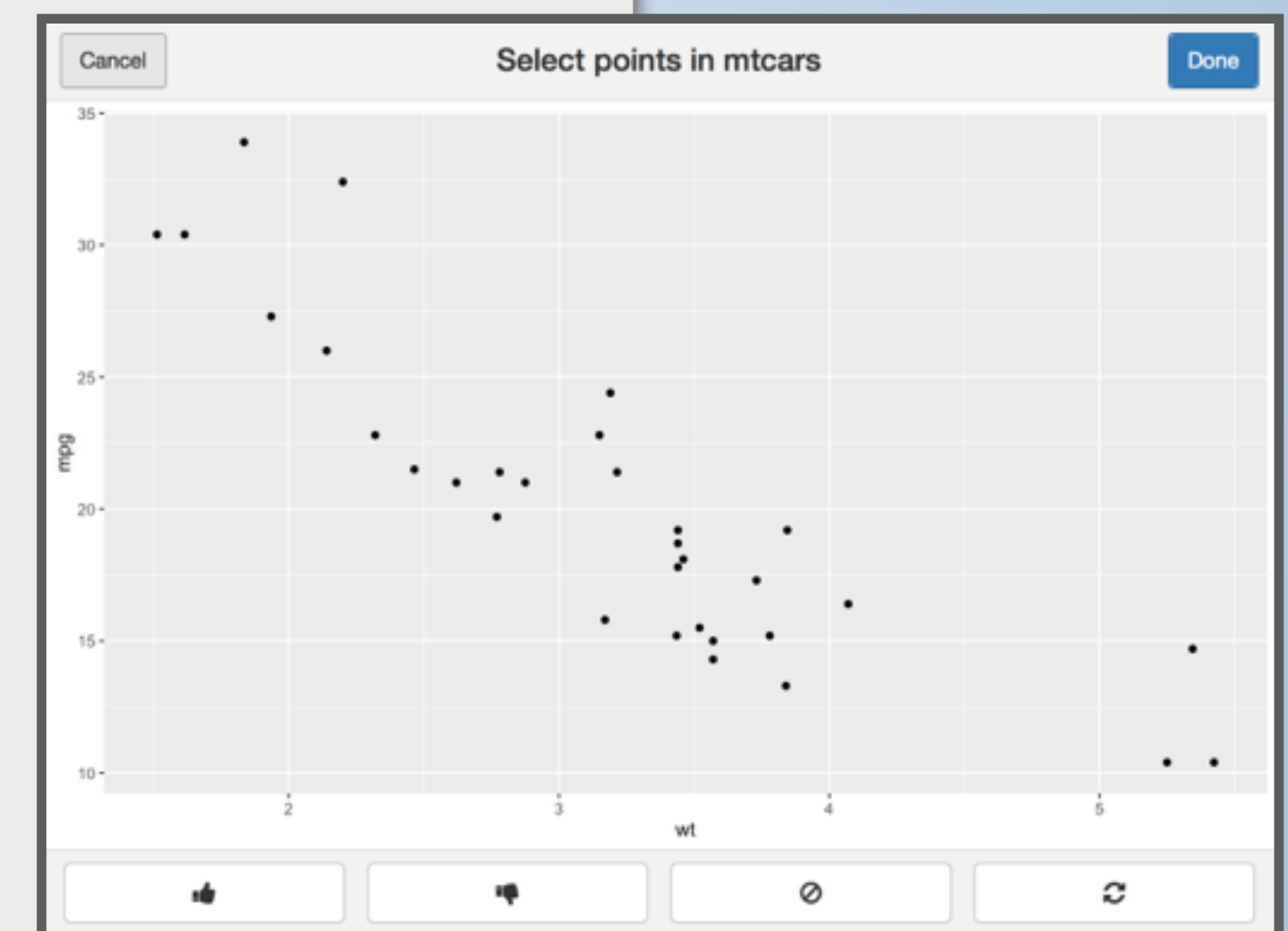
    })

    ...
  }

  runGadget(ui, server)
}

pick_points(mtcars, ~wt, ~mpg)

```



```

pick_points <- function(data, x, y) {
  ...
  server <- function(input, output) {

    vals <- reactiveValues(keep = rep(TRUE, nrow(data)))

    output$plot1 <- renderPlot({
      keep    <- data[ vals$keep, , drop = FALSE]
      exclude <- data[!vals$keep, , drop = FALSE]

      ggplot(keep, aes_(x, y)) +
        geom_point(data = exclude, color = "grey80") +
        geom_point()

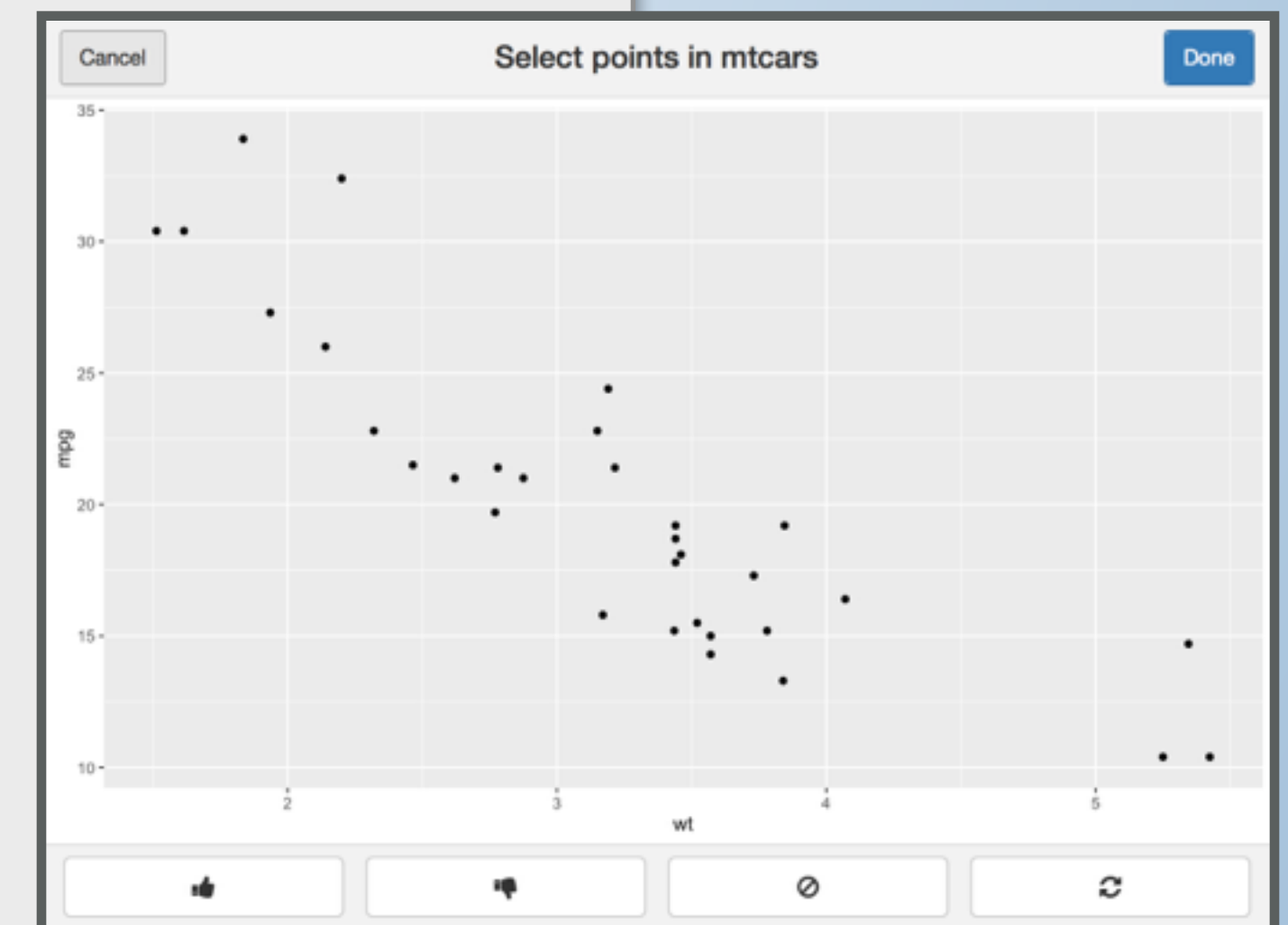
    })

    ...
  }

  runGadget(ui, server)
}

pick_points(mtcars, ~wt, ~mpg)

```





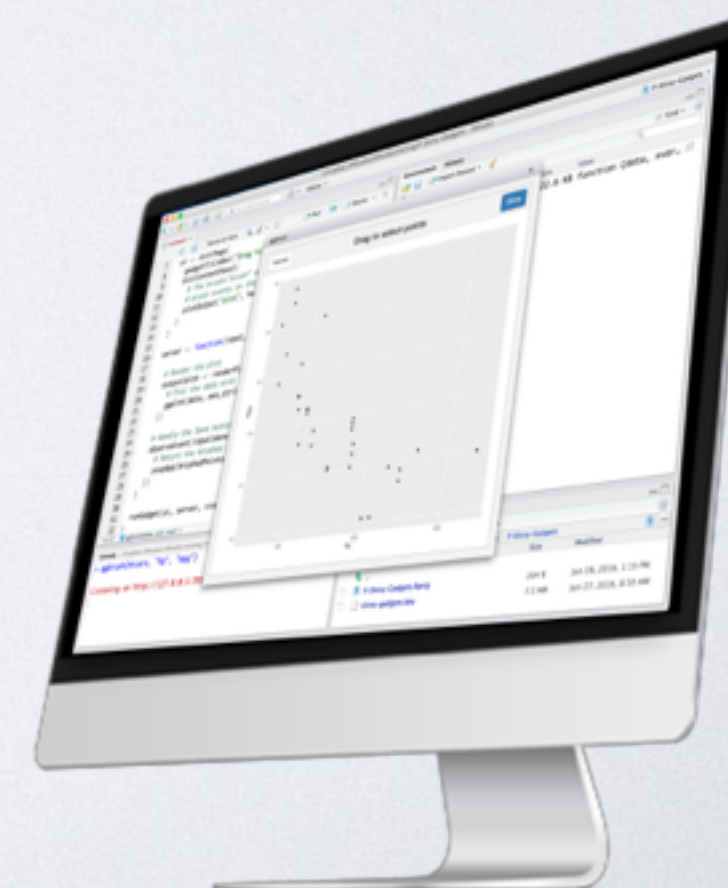
# *runGadget*

Three options for displaying the gadget:

- `runGadget(ui, server, viewer = paneViewer())` (*default*)
- `runGadget(ui, server, viewer = dialogViewer("name"))`
- `runGadget(ui, server, viewer = browserViewer())`



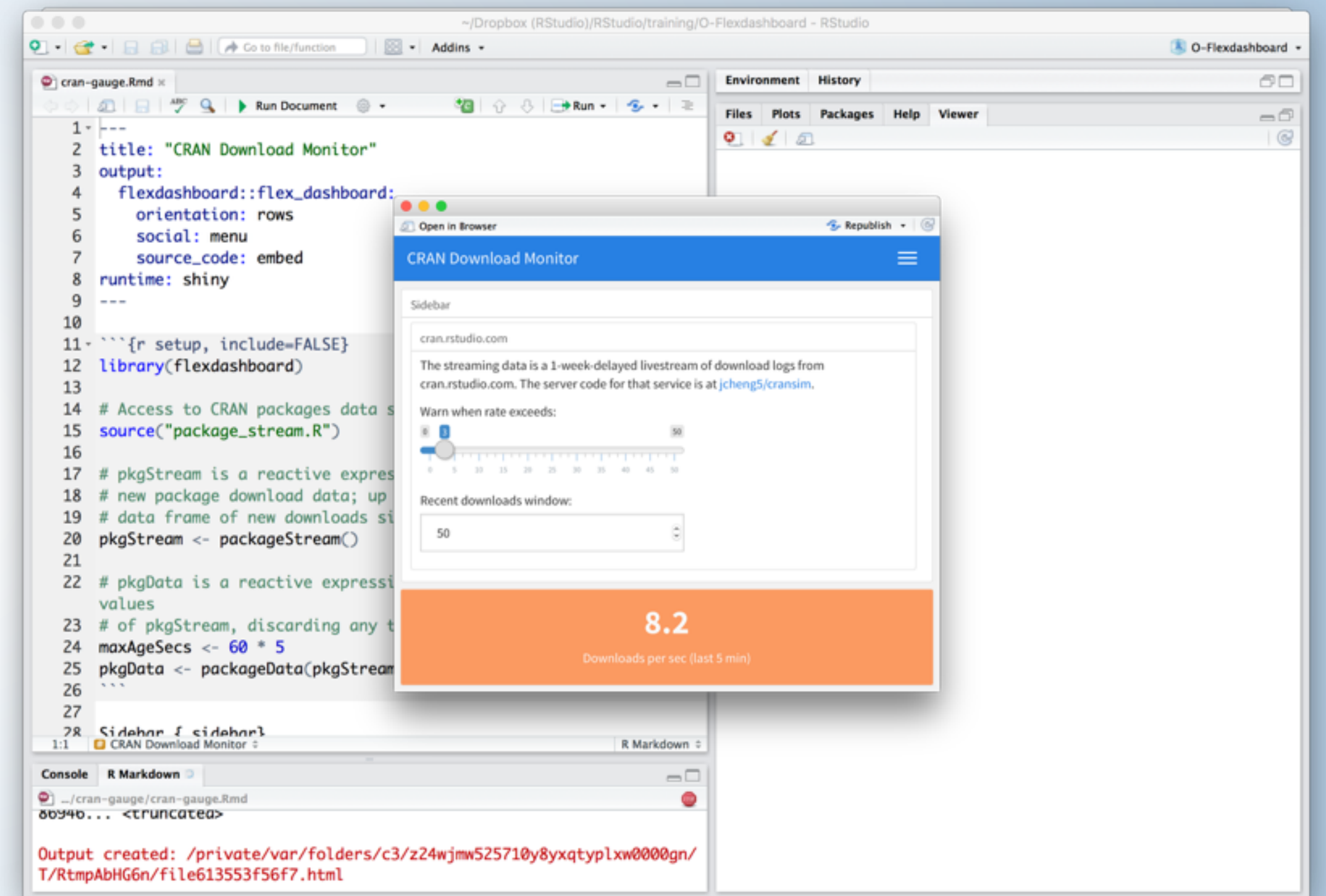
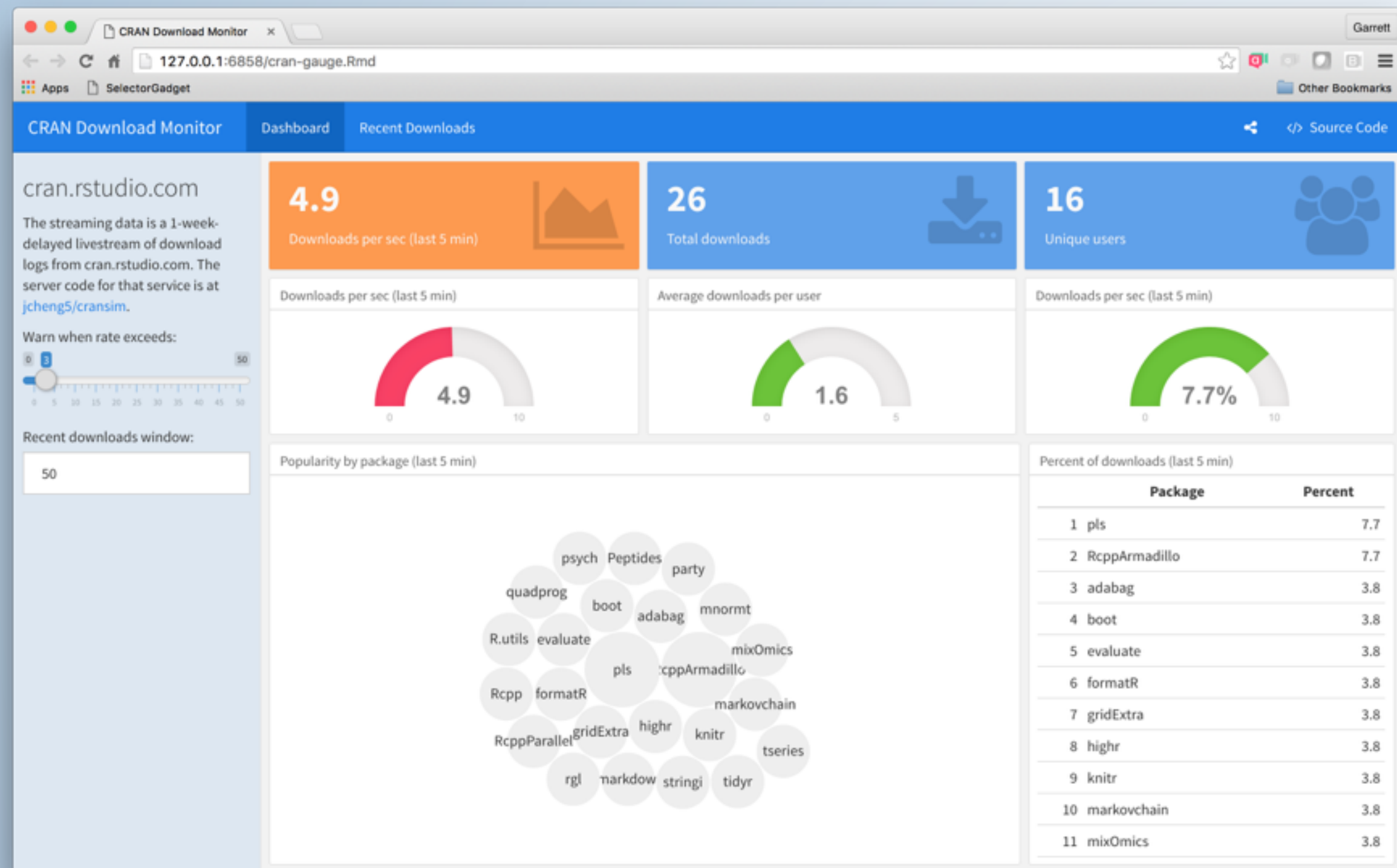
## 2. MINIUI





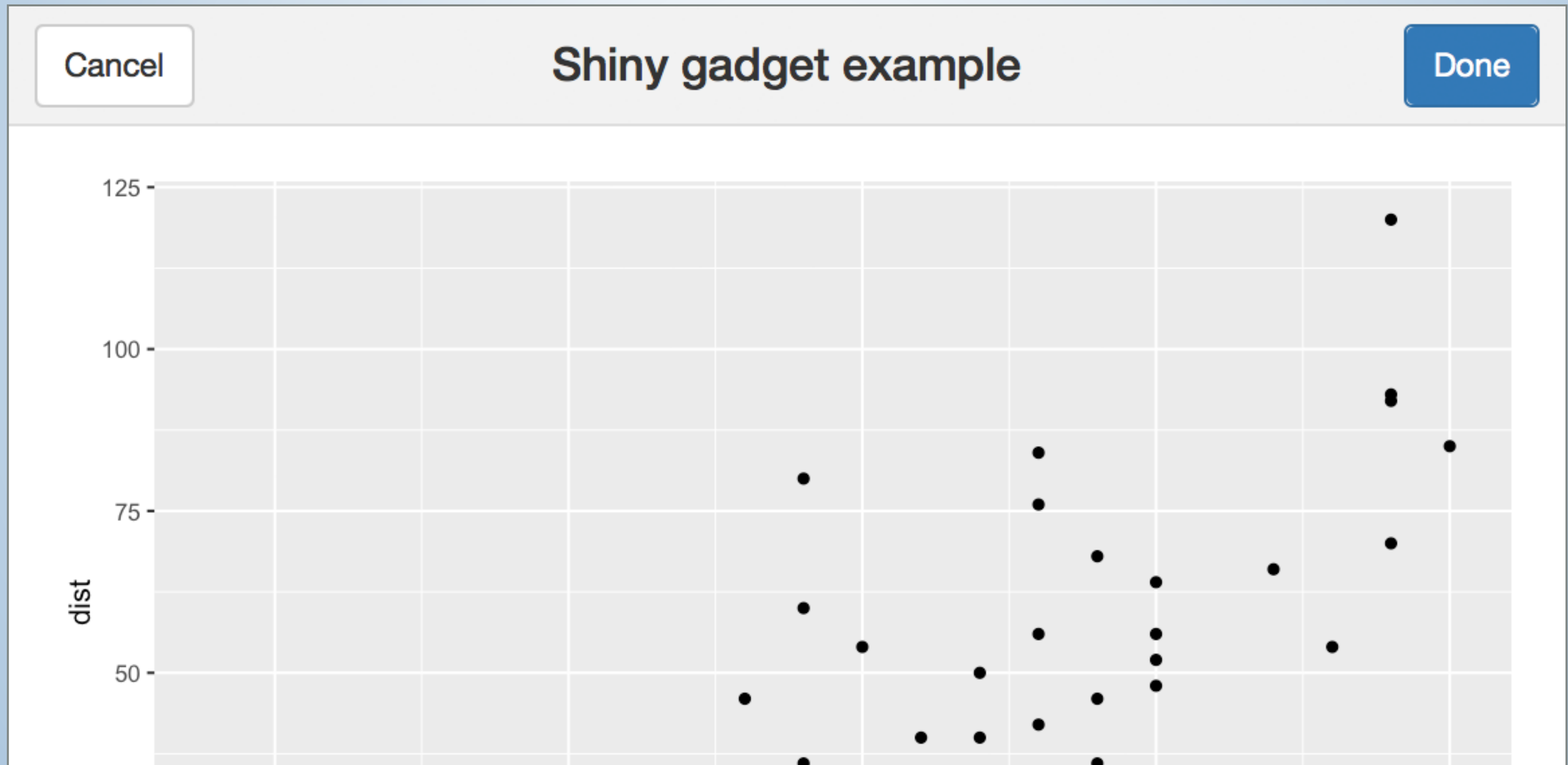
# miniUI

## Shiny UI for small screens



# *title bars*

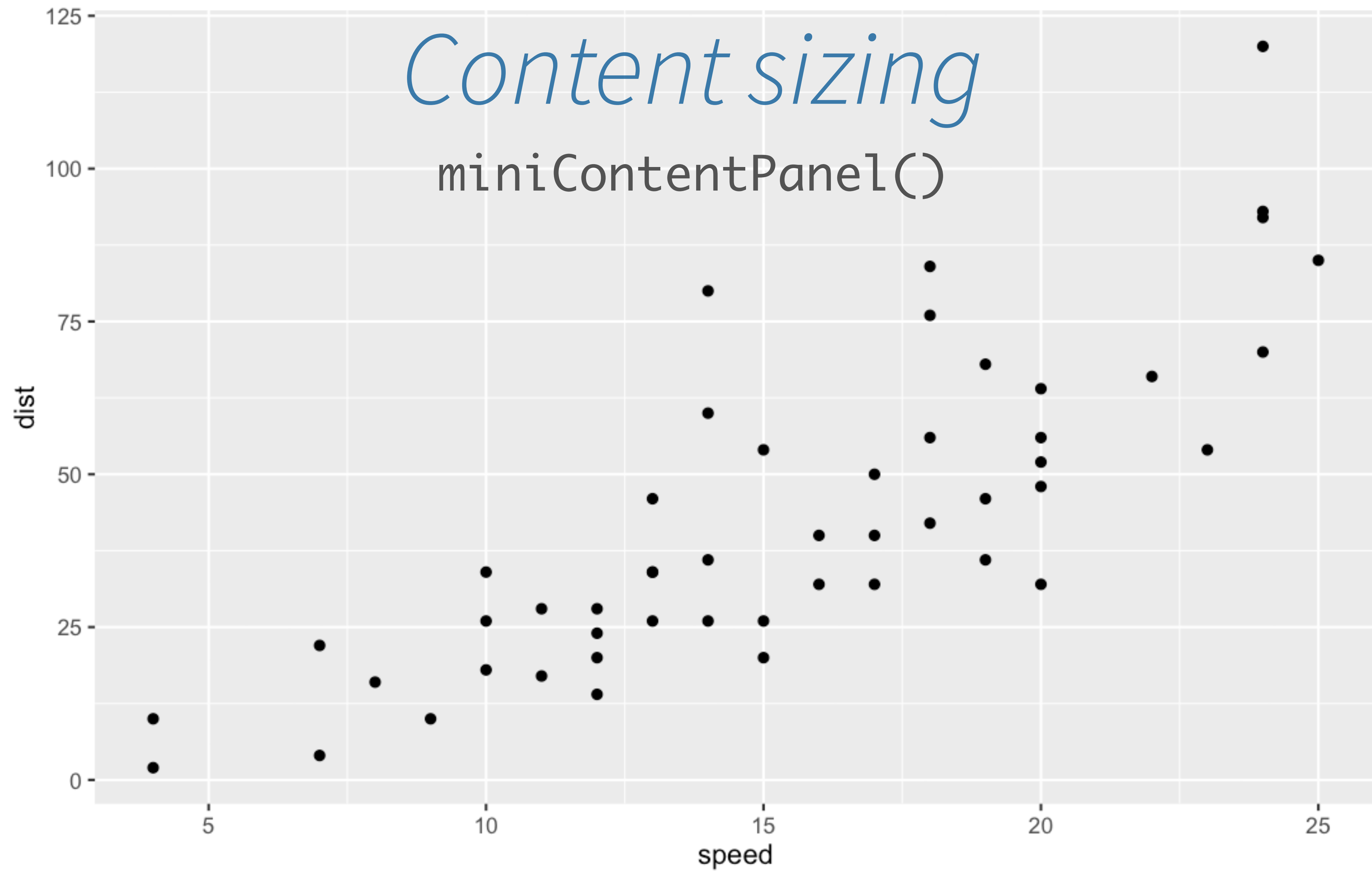
```
gadgetTitleBar("Shiny gadget example")
```

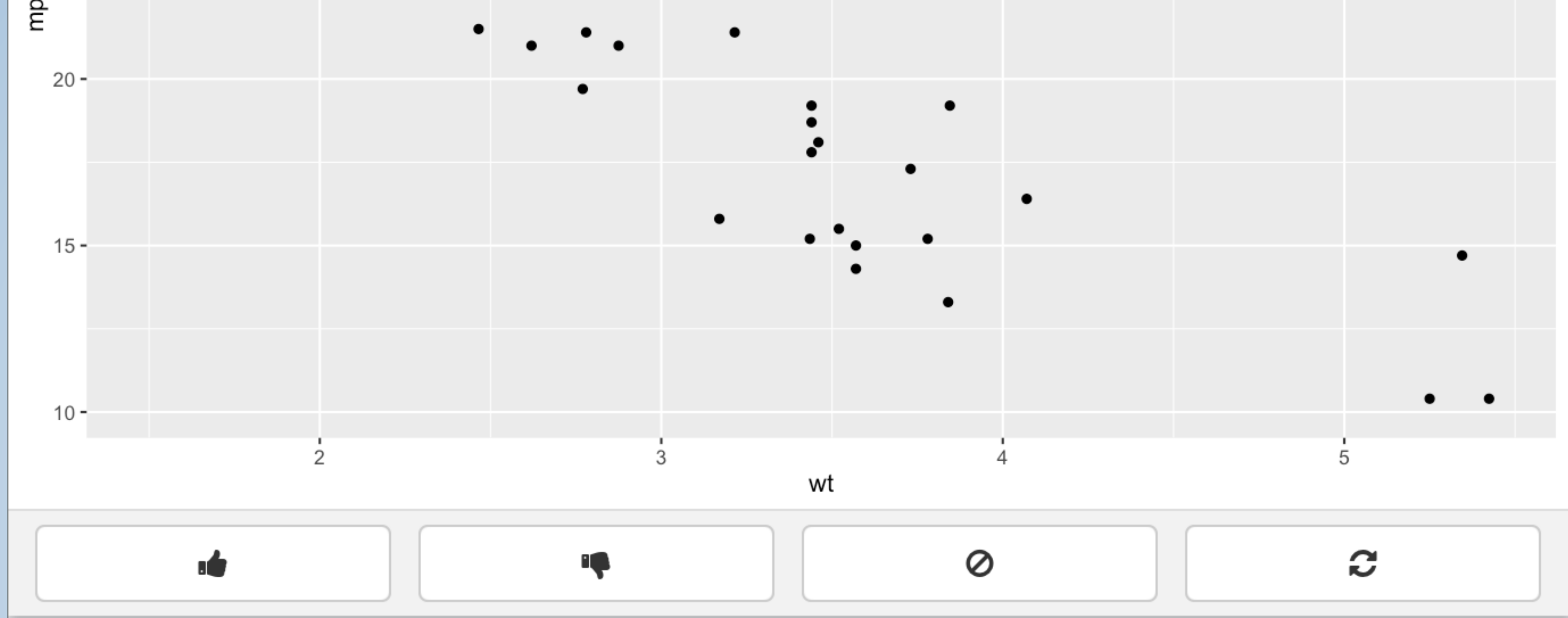




# *Content sizing*

miniContentPanel()

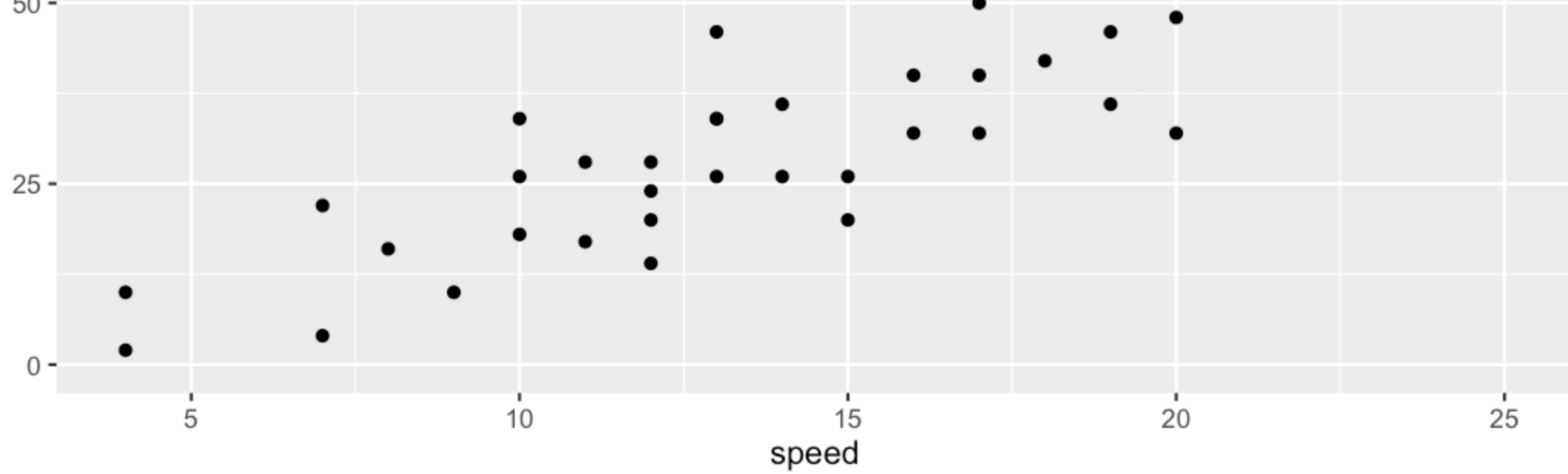





*button block*


miniButtonBlock()






  
Parameters

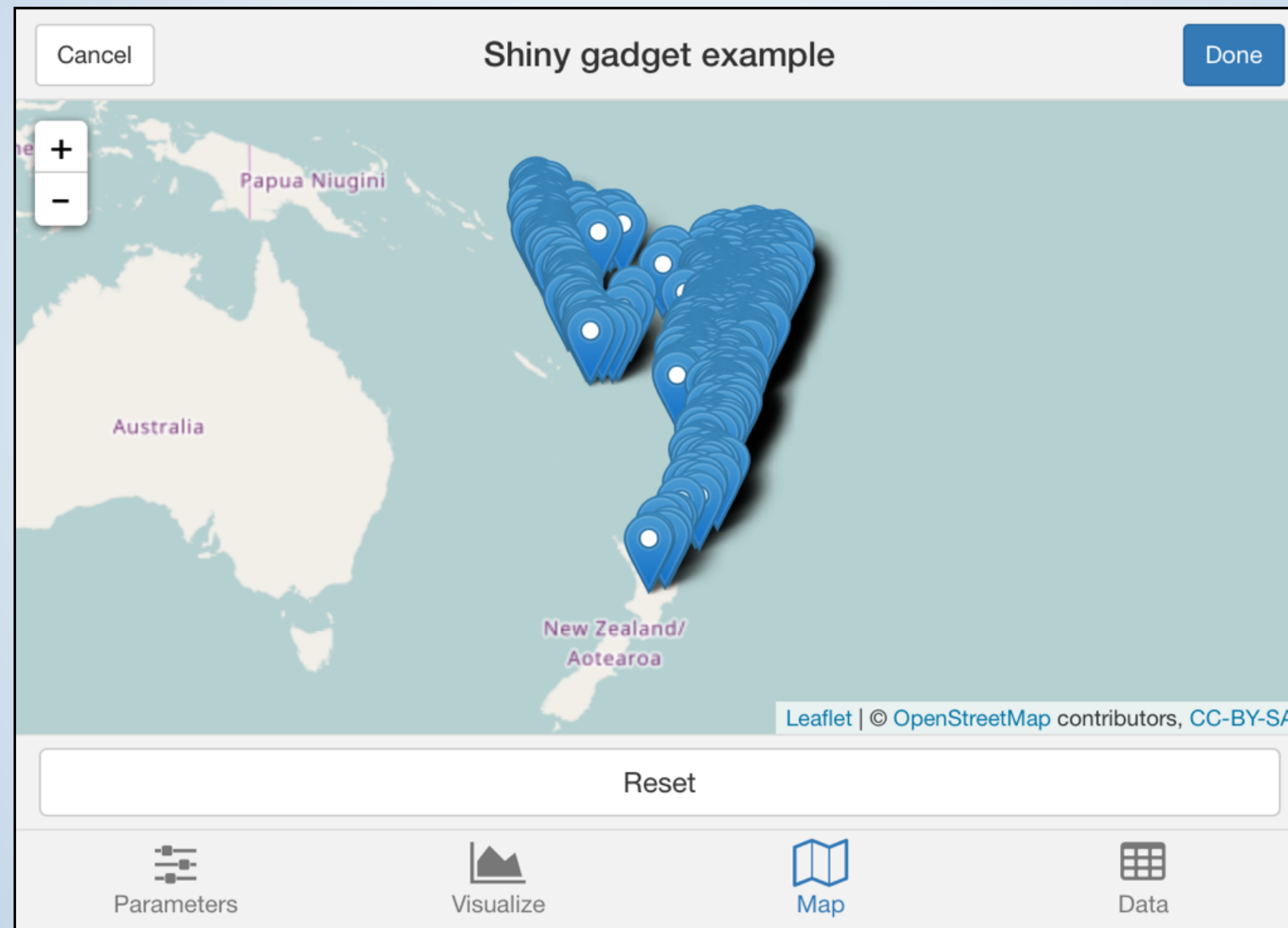
  
Visualize

  
Map

  
Data

*tab strips*  
`miniTabstripPanel()`

# *tabs.R*





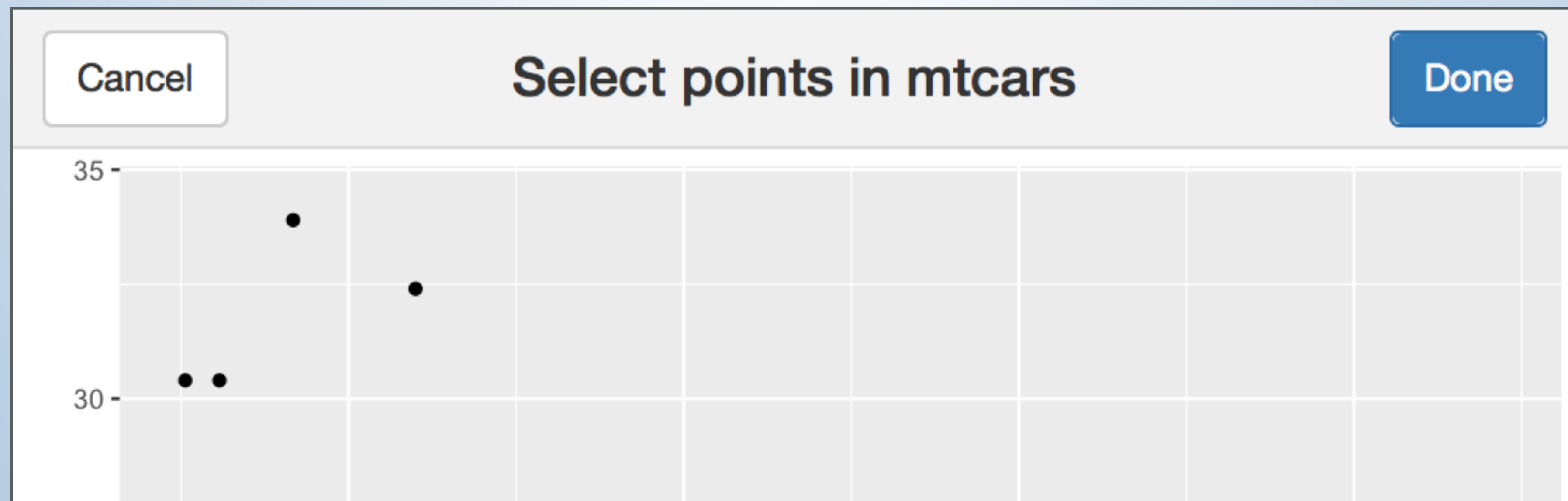
# 3. STOPAPP





# *observeEvent*

```
observeEvent(input$cancel, { stopApp(NULL) })  
observeEvent(input$done, { stopApp(vals$keep) })
```

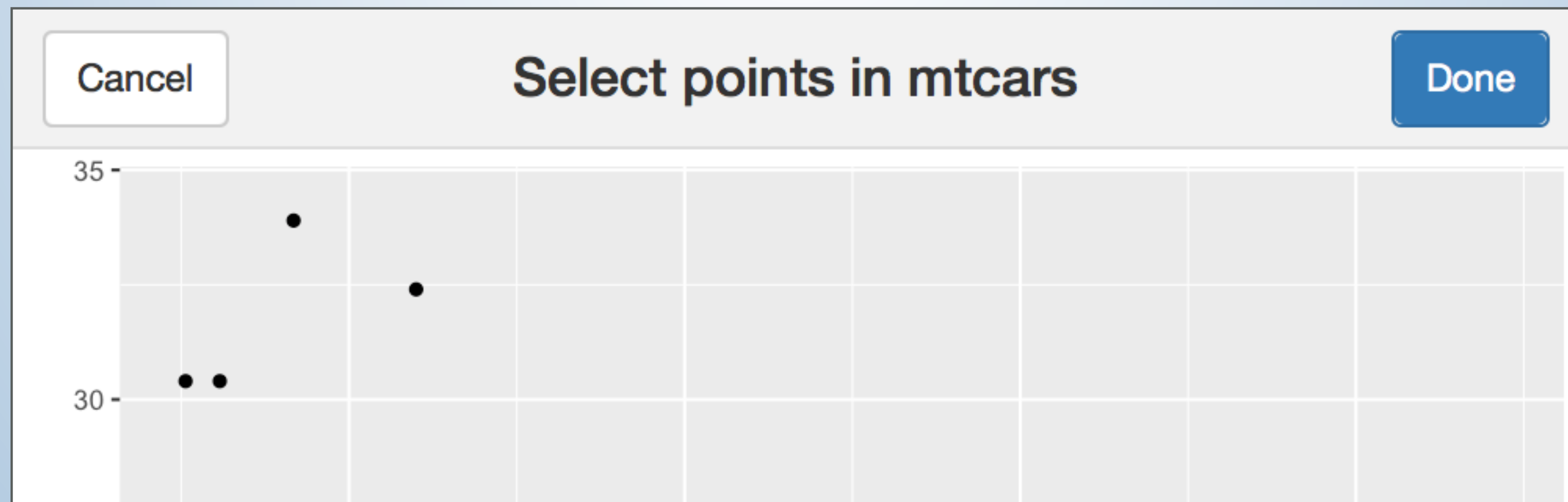




# *observeEvent*

```
observeEvent(input$cancel, { stopApp(NULL) })  
observeEvent(input$done, { stopApp(vals$keep) })
```

When this changes...

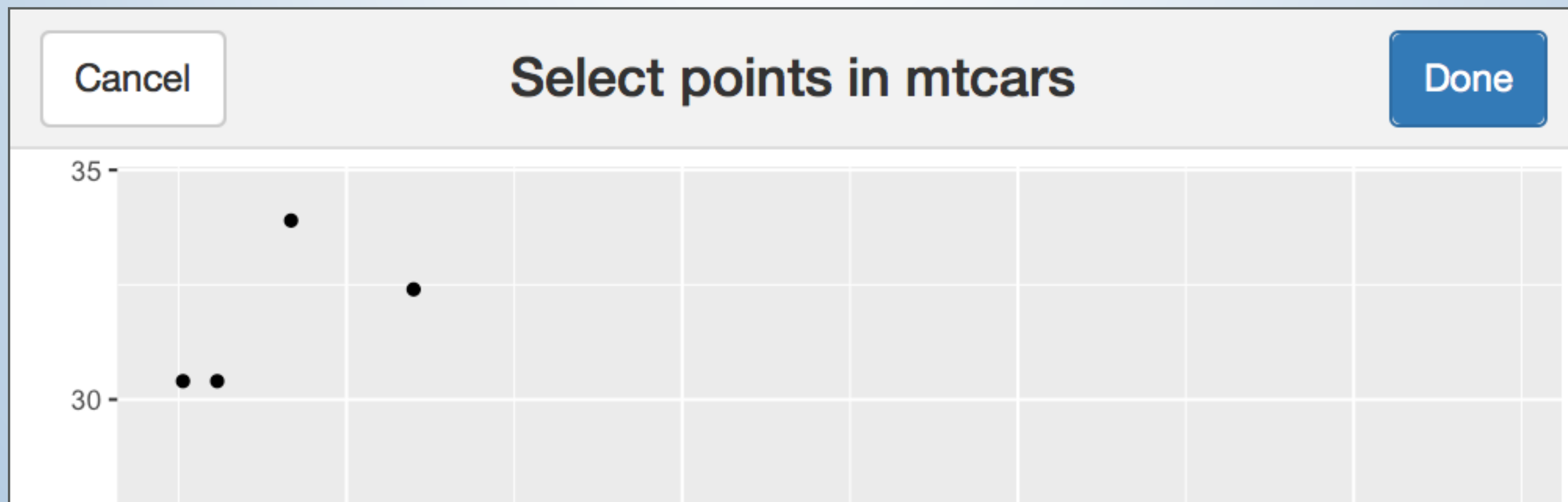


# *observeEvent*

```
observeEvent(input$cancel, { stopApp(NULL) })  
observeEvent(input$done, { stopApp(vals$keep) })
```

When this changes...

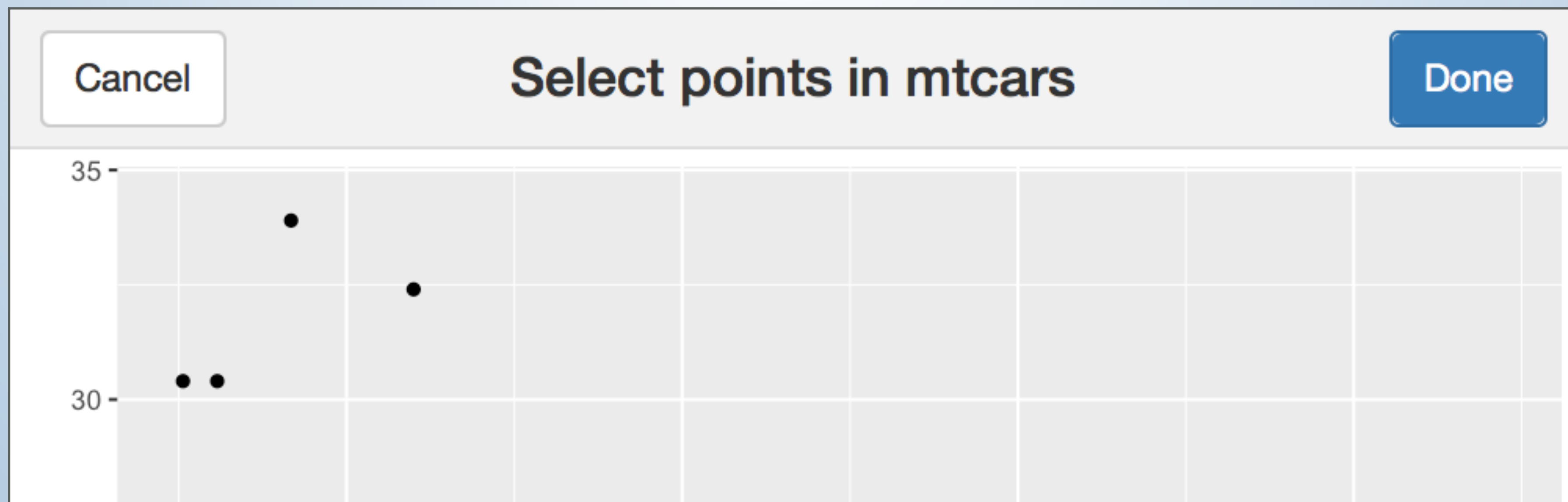
run this.





# *stopApp*

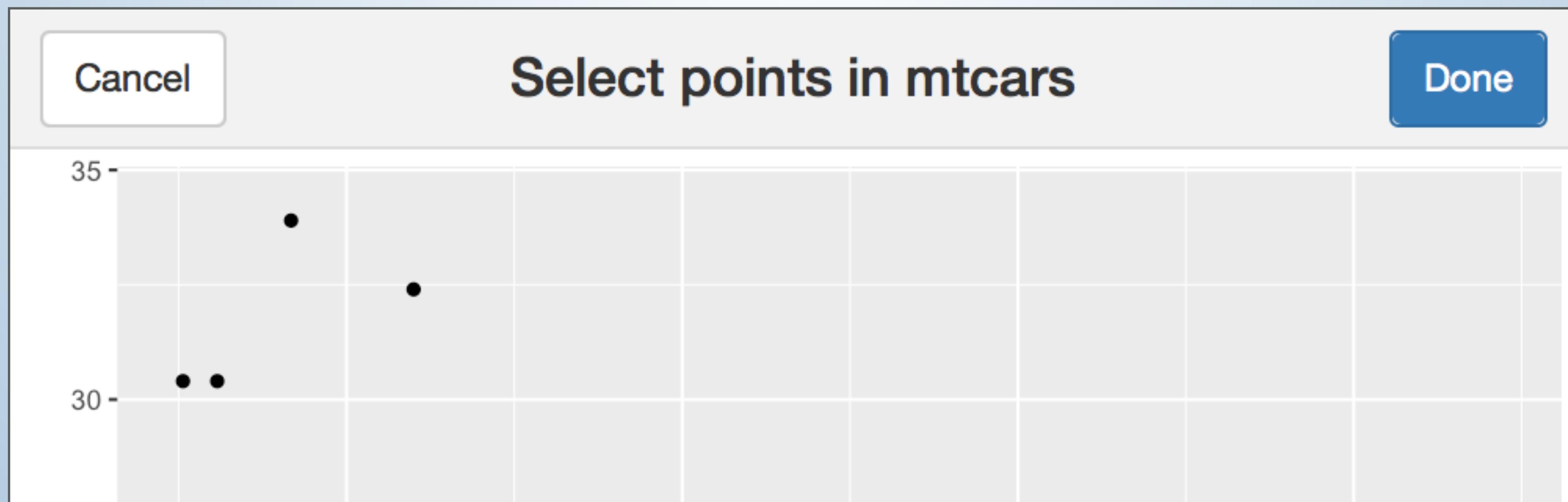
```
observeEvent(input$cancel, { stopApp(NULL) })  
observeEvent(input$done, { stopApp(vals$keep) })
```



# *stopApp*

```
observeEvent(input$cancel, { stopApp(NULL) })  
observeEvent(input$done, { stopApp(vals$keep) })
```

Exit the app



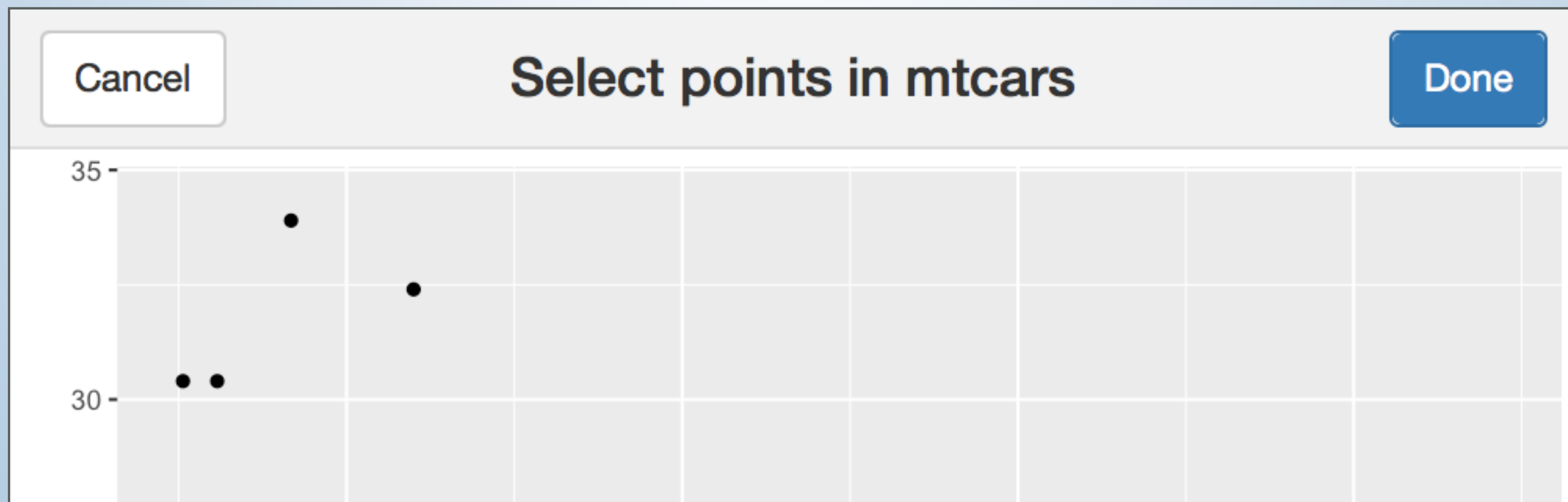


# *stopApp*

```
observeEvent(input$cancel, { stopApp(NULL) })  
observeEvent(input$done, { stopApp(vals$keep) })
```

Exit the app

Return this value





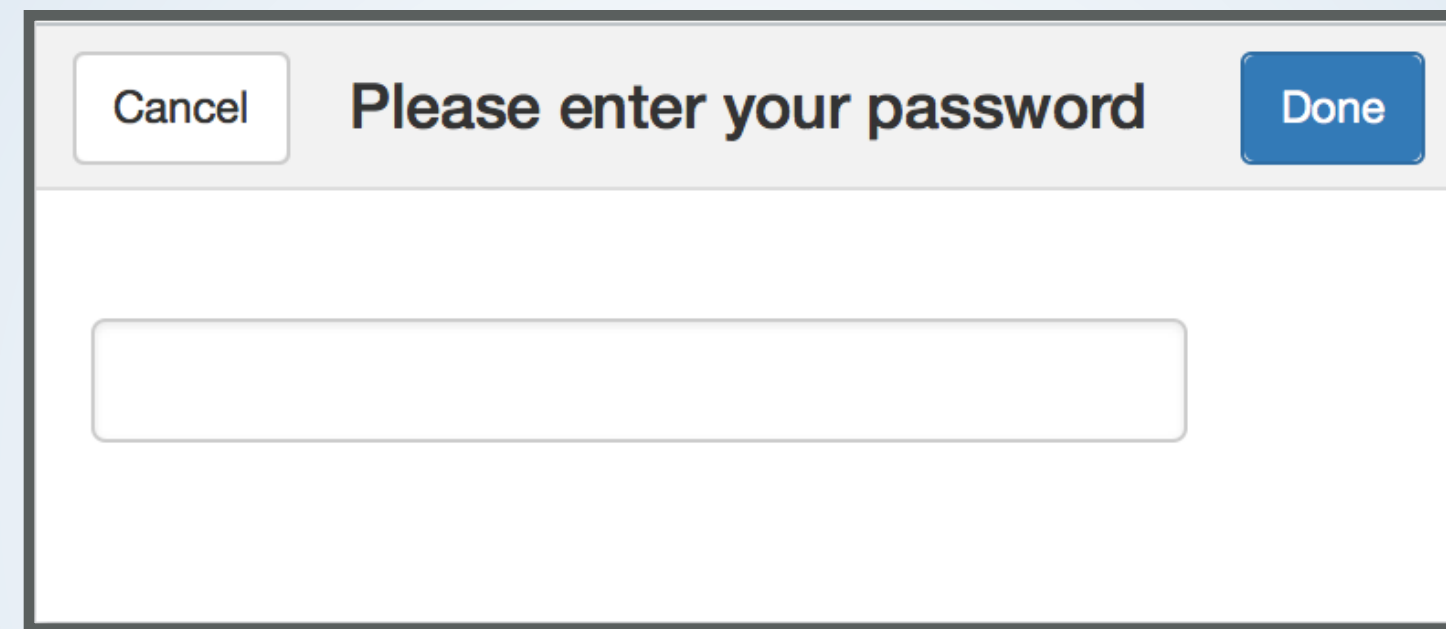
# ADDINS





# *3 Essential Features*

2. `library(miniUI)`



A screenshot of a password prompt dialog box. The dialog has a title bar with the text "Please enter your password". Below the title bar, there are two buttons: "Cancel" on the left and "Done" on the right. In the center of the dialog is a text input field for entering a password.

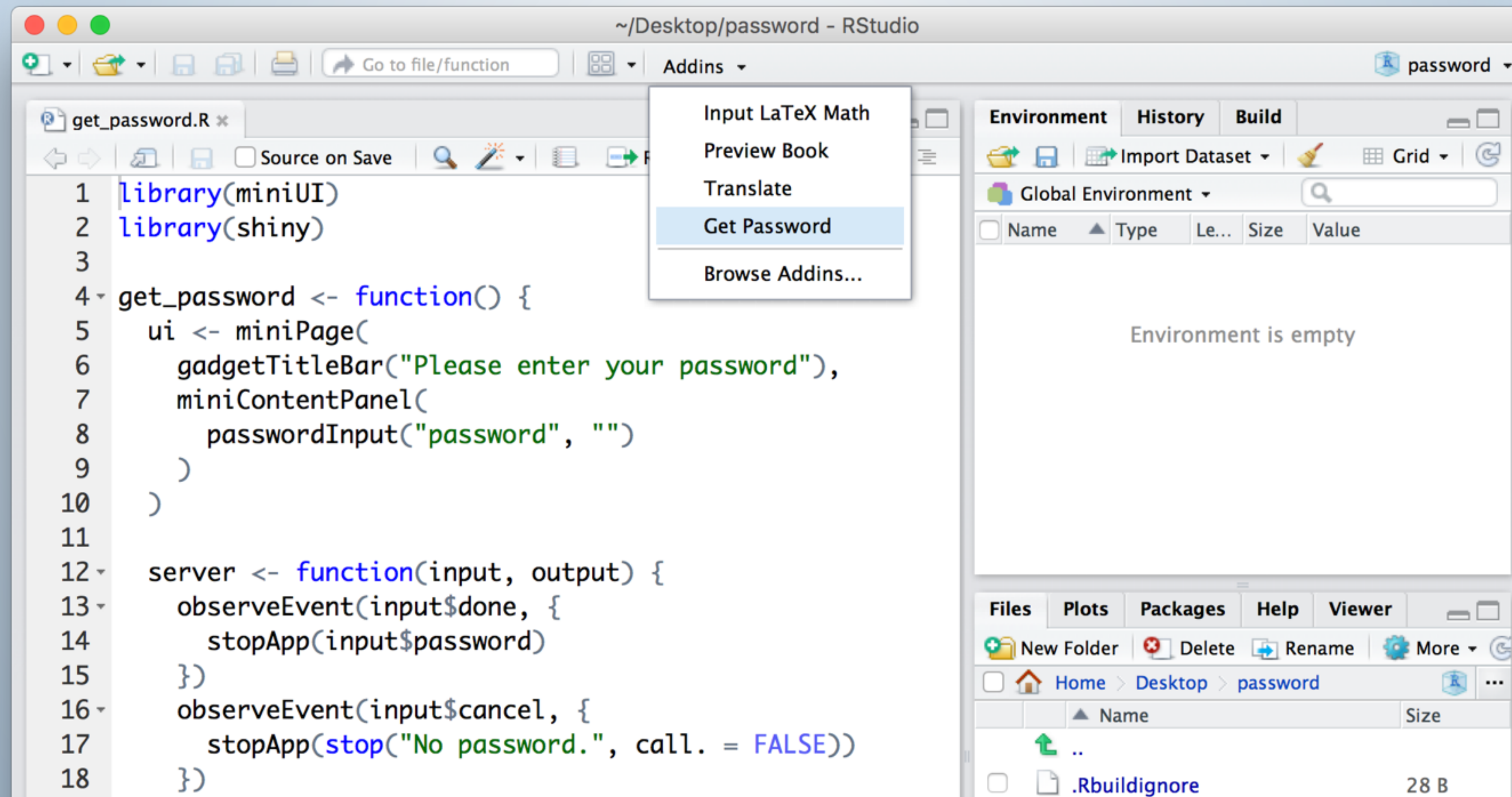
1. `get_password()`

3. `stopApp()`

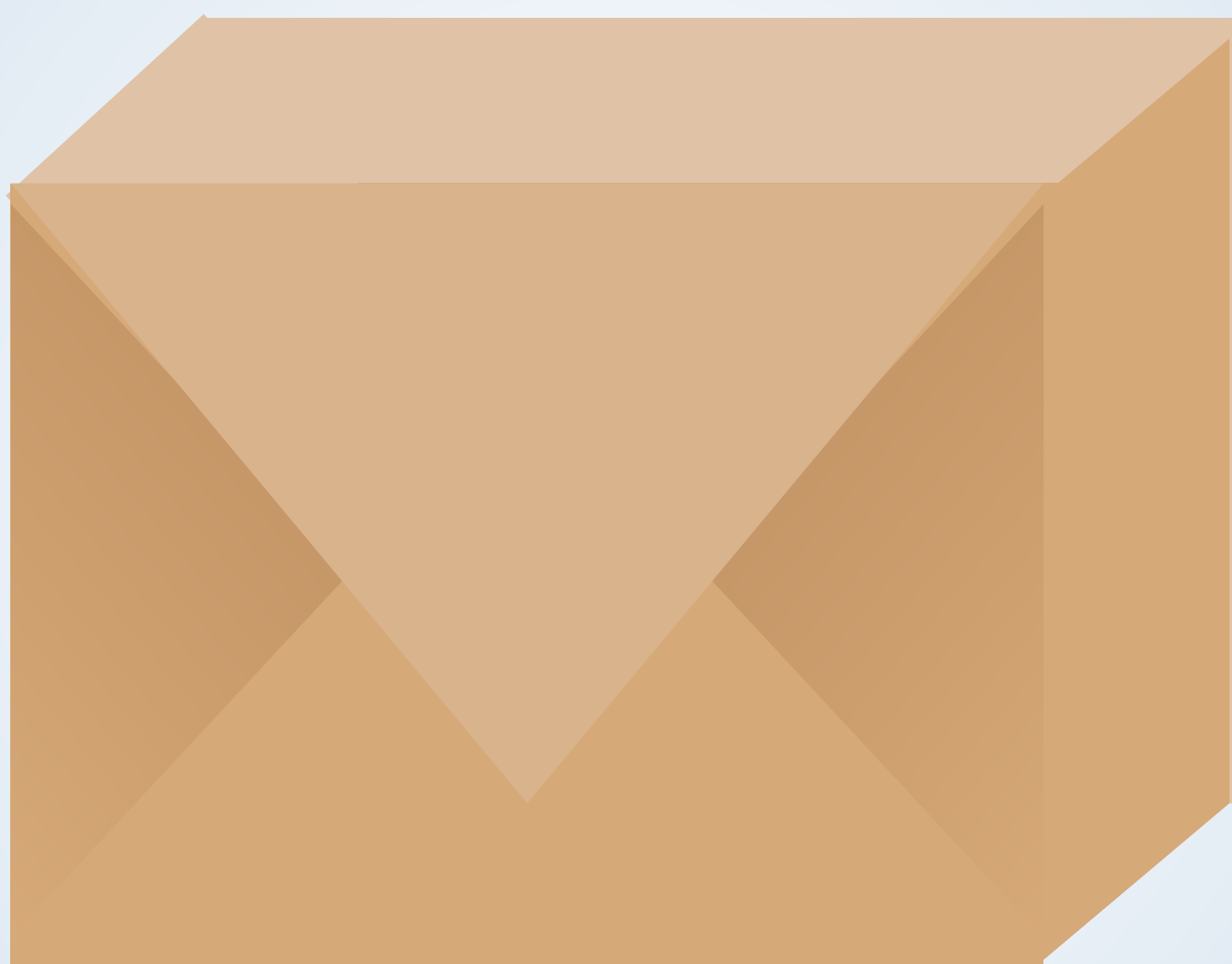
Programming Workflow

# Addins

A function that you can call straight for the RStudio IDE GUI or a keyboard shortcut







Save add in as a function. In package, add `inst/rstudio/addins.dcf`:

```
Name: Get Password 2  
Description: Collects password  
Binding: get_password  
Interactive: true
```

[rstudio.github.io/rstudioaddins/](https://rstudio.github.io/rstudioaddins/)



# *ideas*

- **addinexamples** - package of add ins
- **hadladdin** - demo add ins by Hadley Wickham
- **rstudioapi** - package of functions to manipulate the RStudio IDE



# CONCLUSION

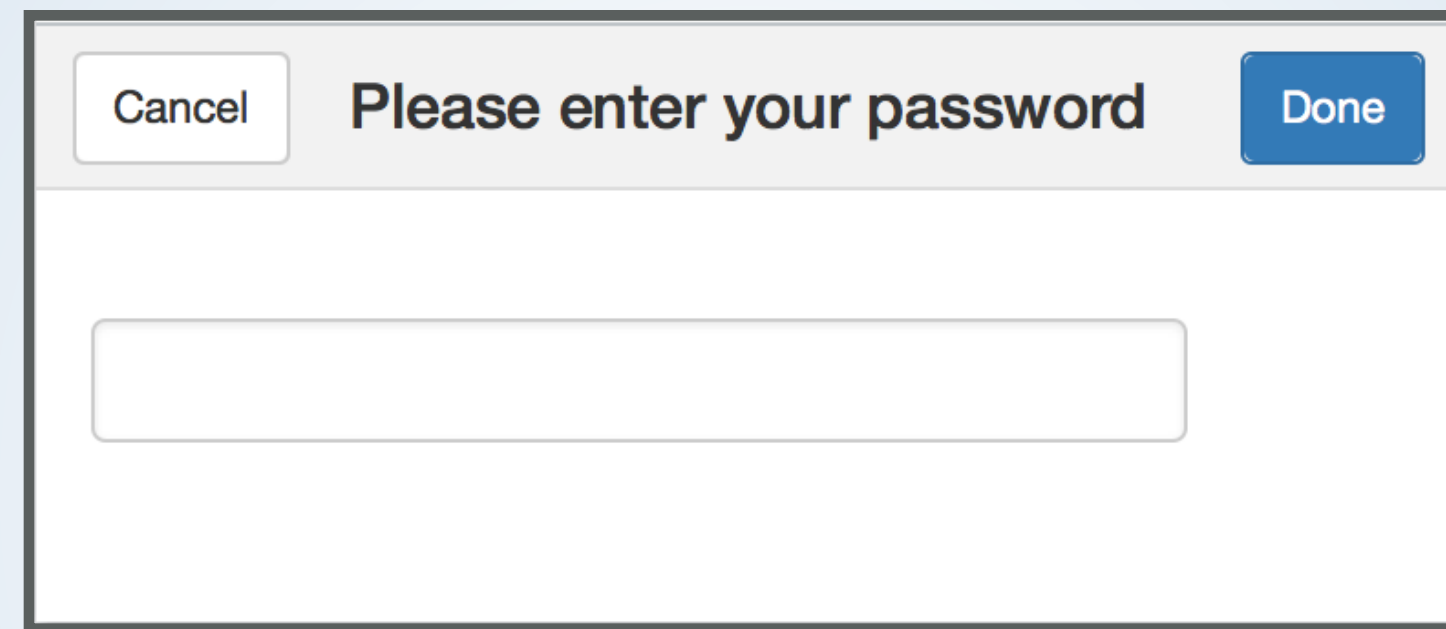




# *To make a gadget, remember 3 things:*

2. `library(miniUI)`

1. `get_password()`



A screenshot of a password prompt dialog box. The dialog has a title bar with the text "Please enter your password". Below the title bar is a text input field. At the bottom of the dialog are two buttons: "Cancel" on the left and "Done" on the right.

3. `stopApp()`

Programming Workflow



# THANK YOU

[shiny.rstudio.com/articles/gadgets.html](https://shiny.rstudio.com/articles/gadgets.html)

[shiny.rstudio.com/articles/gadget-ui.html](https://shiny.rstudio.com/articles/gadget-ui.html)

[rstudio.github.io/rstudioaddins/](https://rstudio.github.io/rstudioaddins/)

