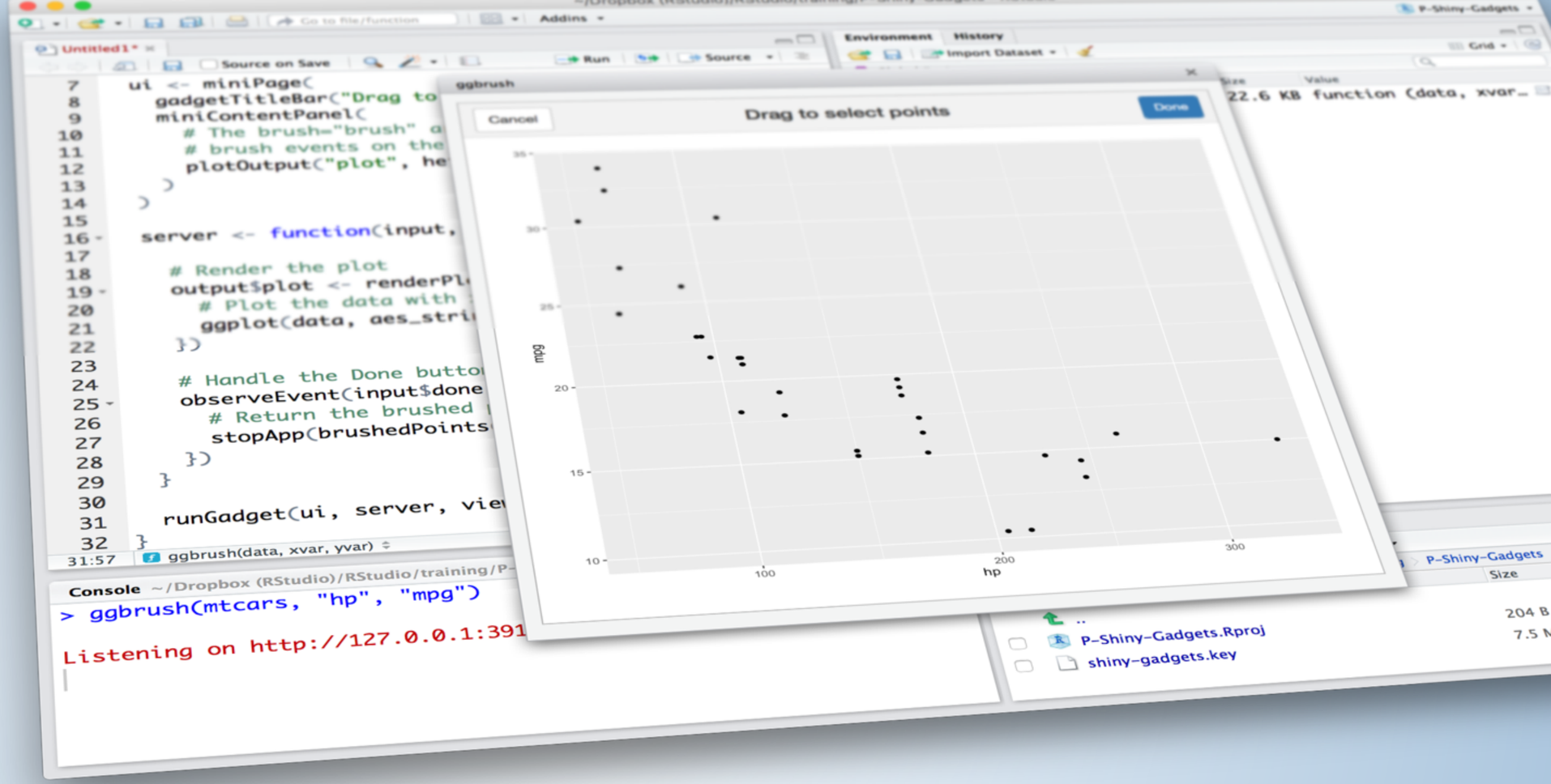


SHINY GADGETS

INTERACTIVE TOOLS FOR PROGRAMMING

RStudio

WHAT IS A GADGET?

Shiny app

Shiny gadget

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?

Leverage analysis

Data cleaner

Model/simulation UI

Color picker

Plot builder

Data store UI

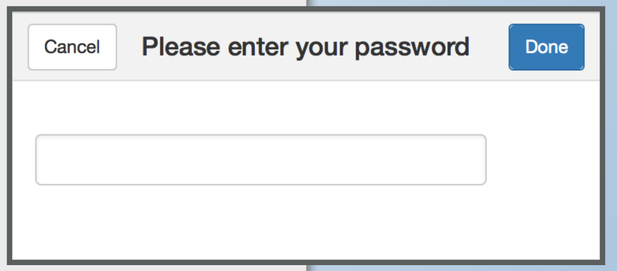
Data filterer

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

get_password.R

3 Essential Features

```r
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()
```



```r
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

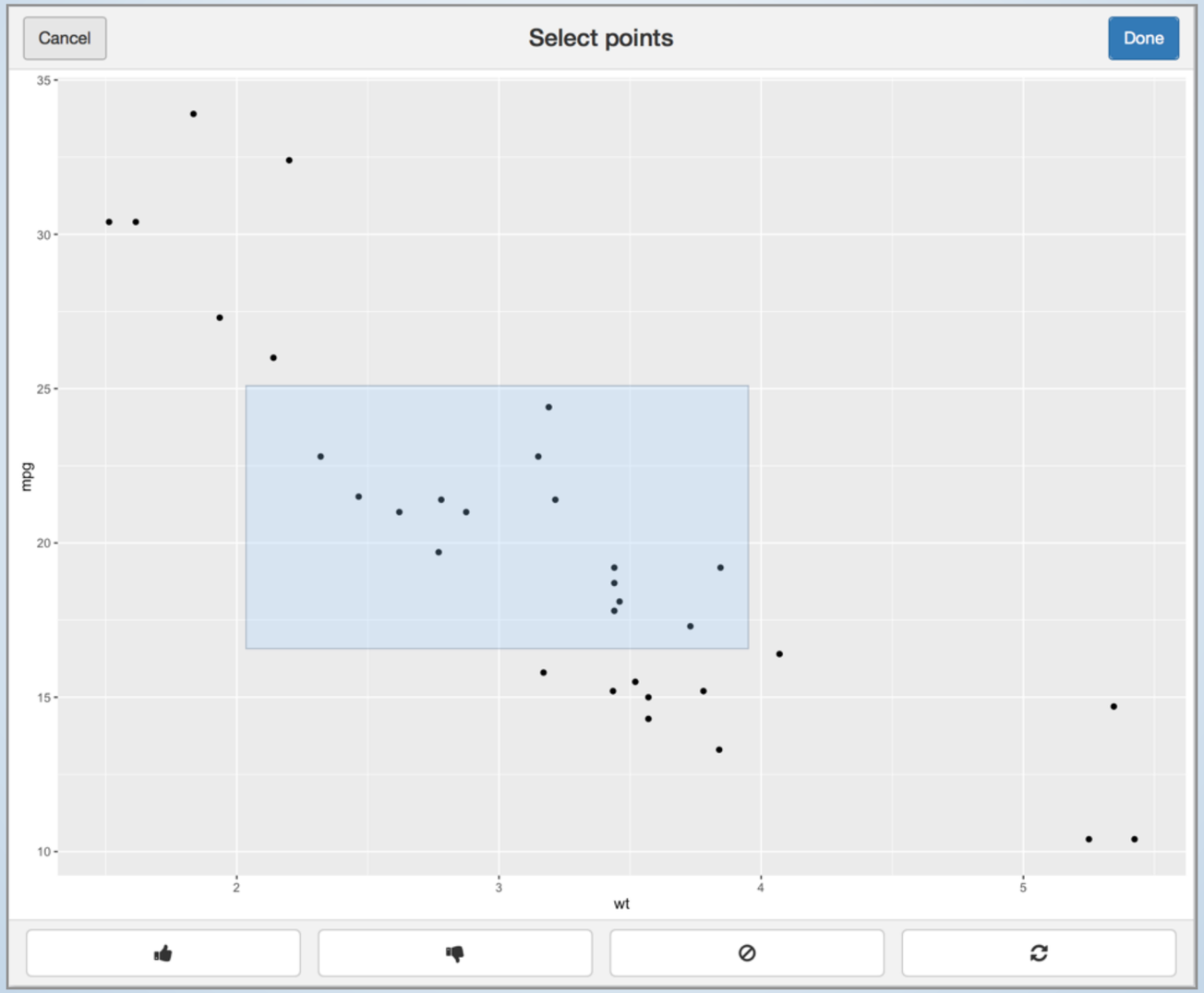
```
get_password <- function() {
  ui <- miniPage(
    gadgetTitleBar("Please enter your password"),
    miniContentPanel(
      passwordInput("password", "")
    )
  )

  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()
```

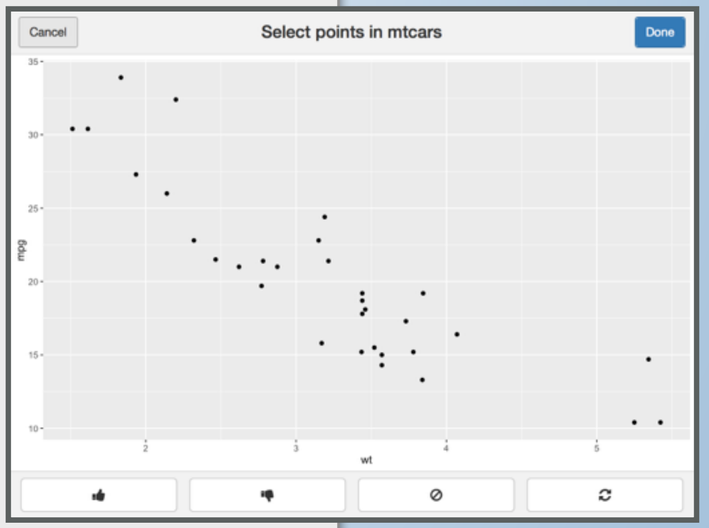
3 **Close app, return values with stopApp()**

brush_gadget.R

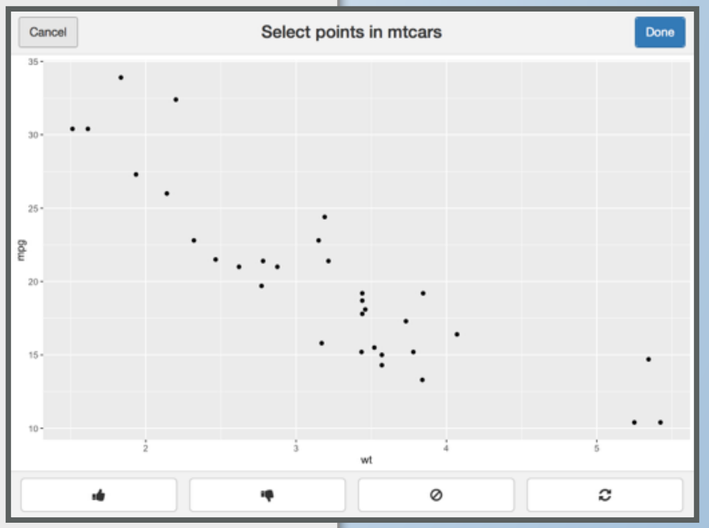


1. <– FUNCTION

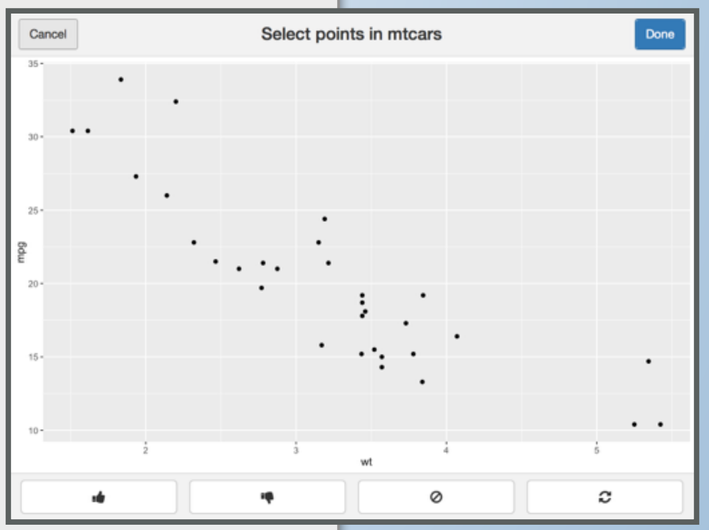
```r
pick_points <- function(data, x, y) {
  …
  server <- function(input, output) {

    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)
```



```r
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)
```



```r
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)
```

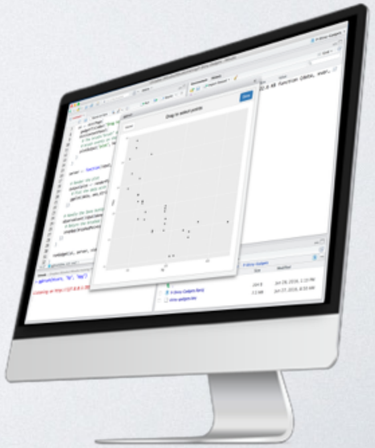


runGadget

Three options for displaying the gadget:

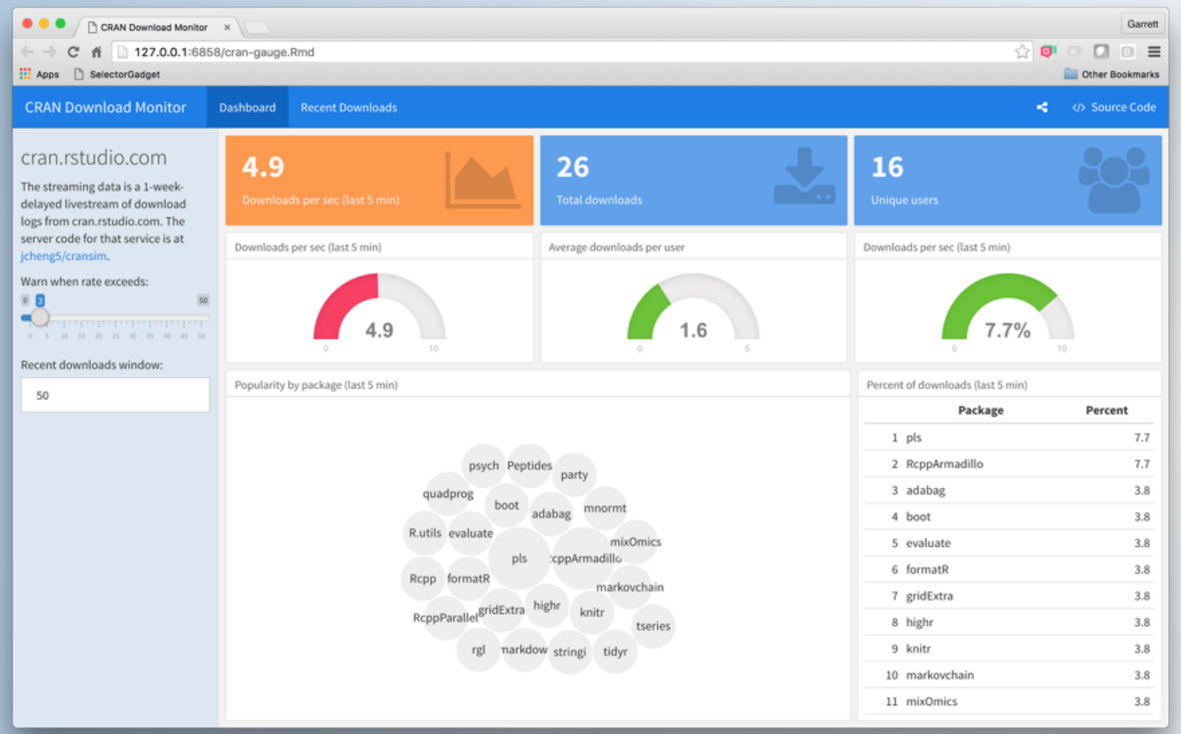
- `runGadget(ui, server, viewer = paneViewer())` *(default)*
- `runGadget(ui, server, viewer = dialogViewer("name"))`
- `runGadget(ui, server, viewer = browserViewer())`

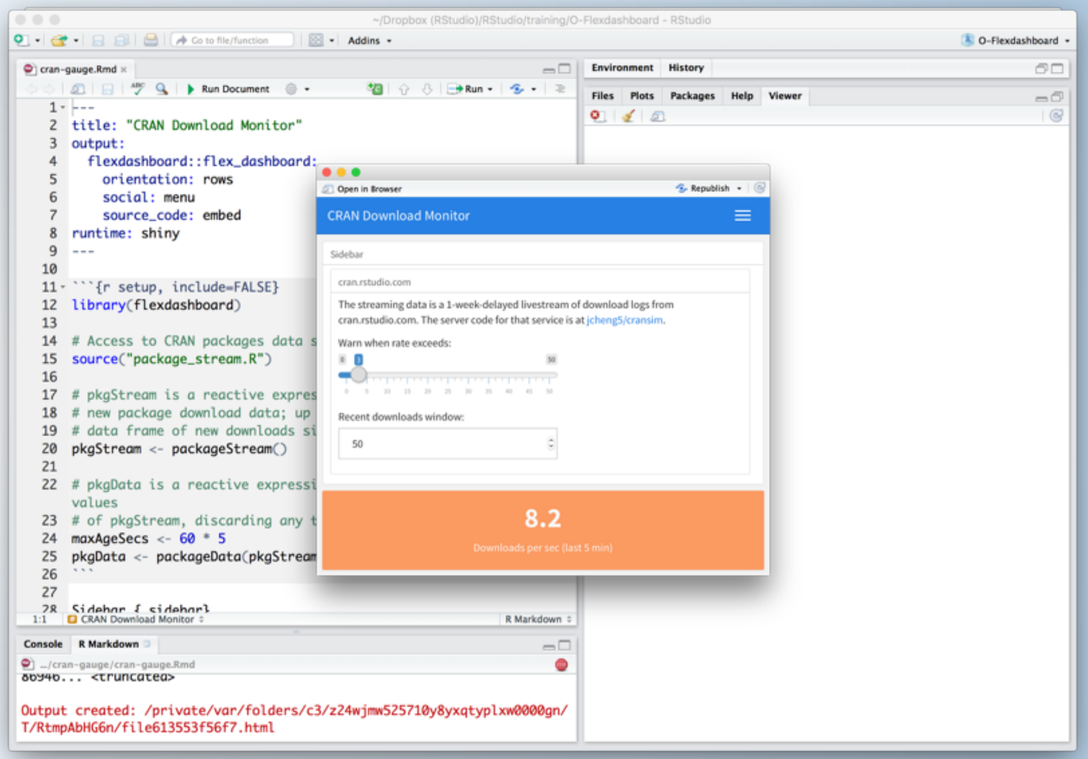
2. MINIUI



miniUI

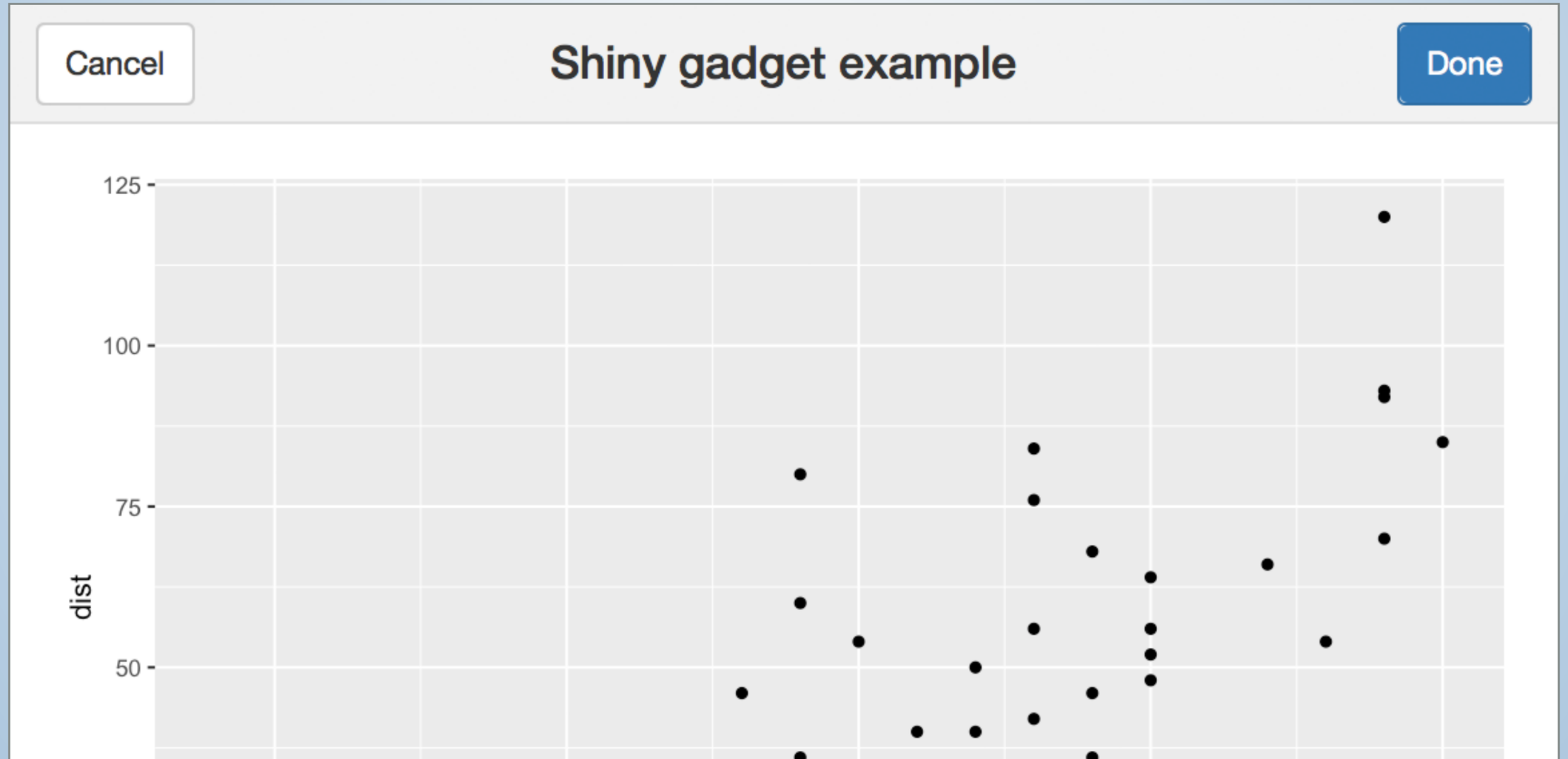
Shiny UI for small screens





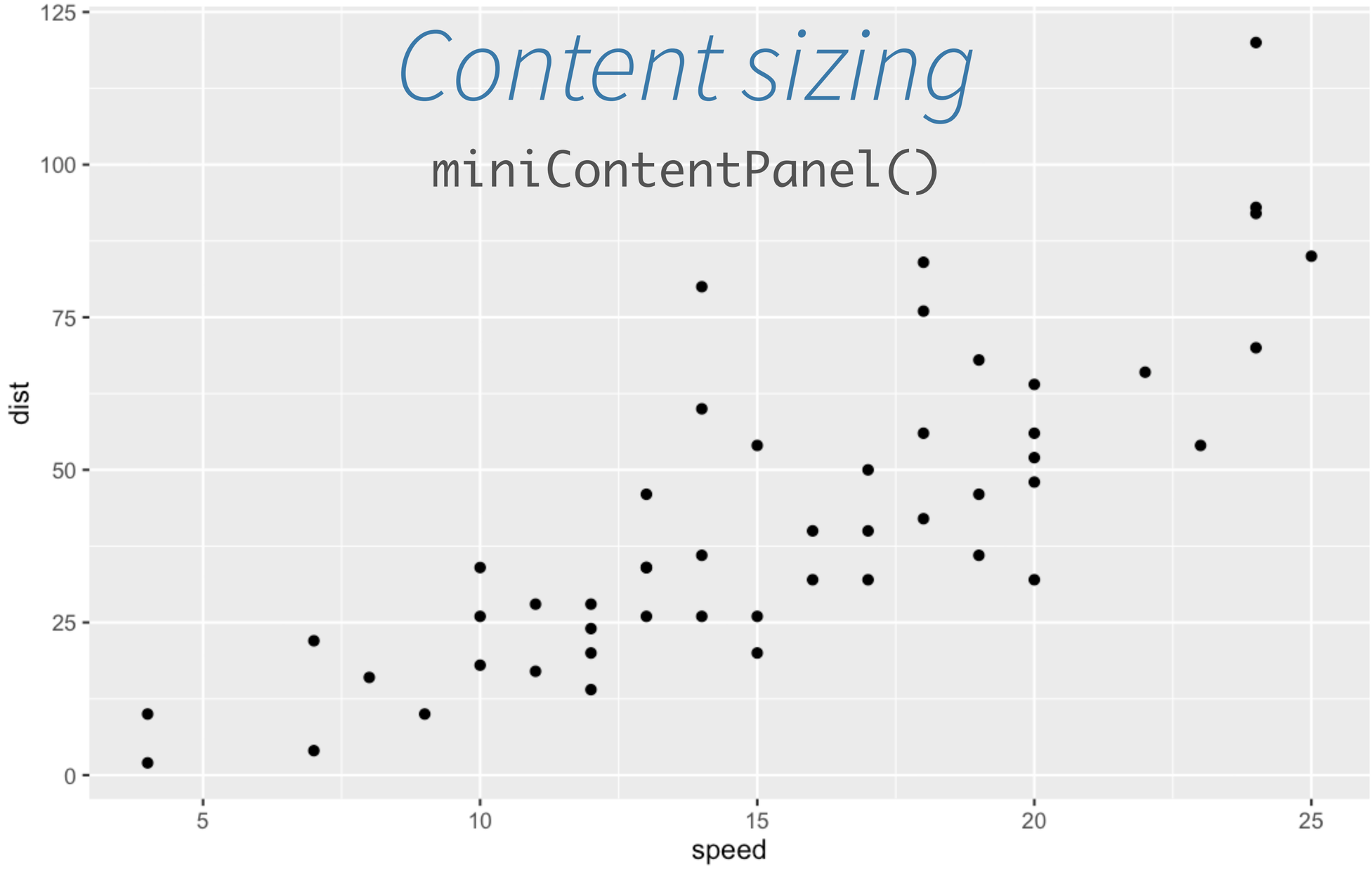
title bars

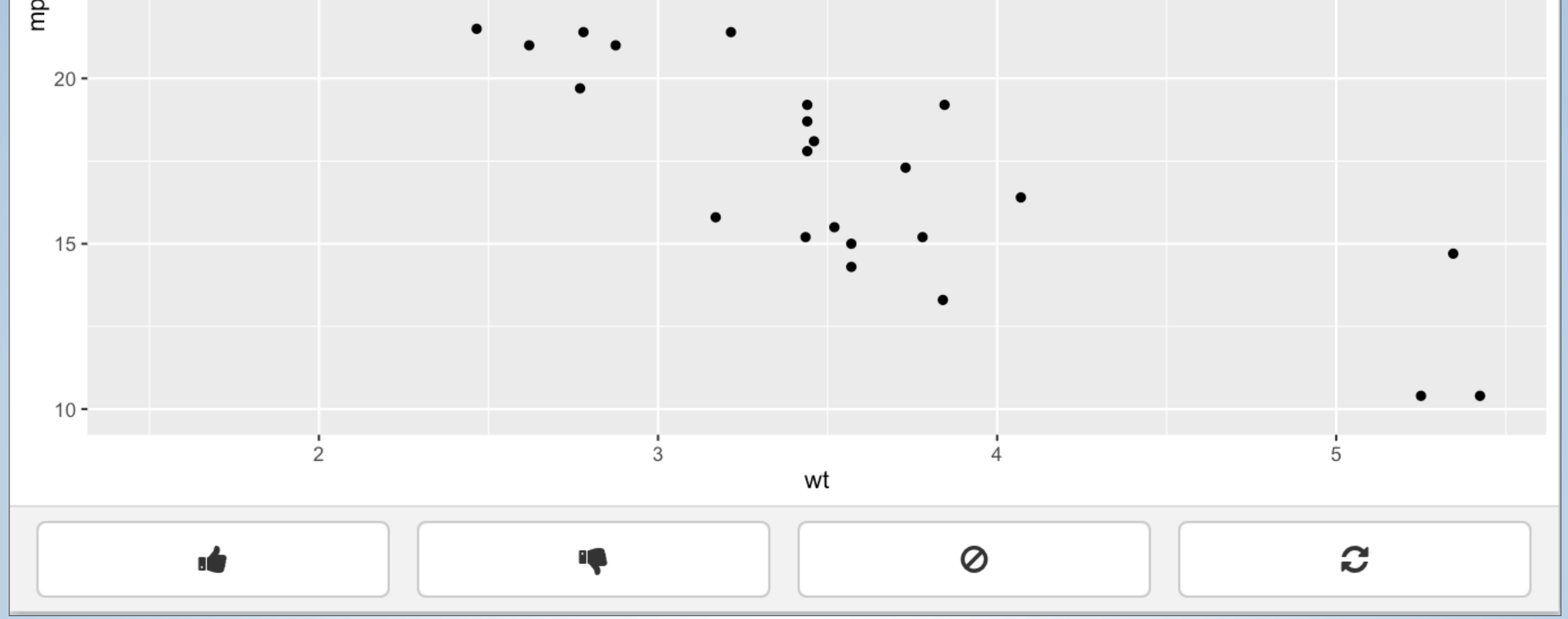
```
gadgetTitleBar("Shiny gadget example")
```



Content sizing

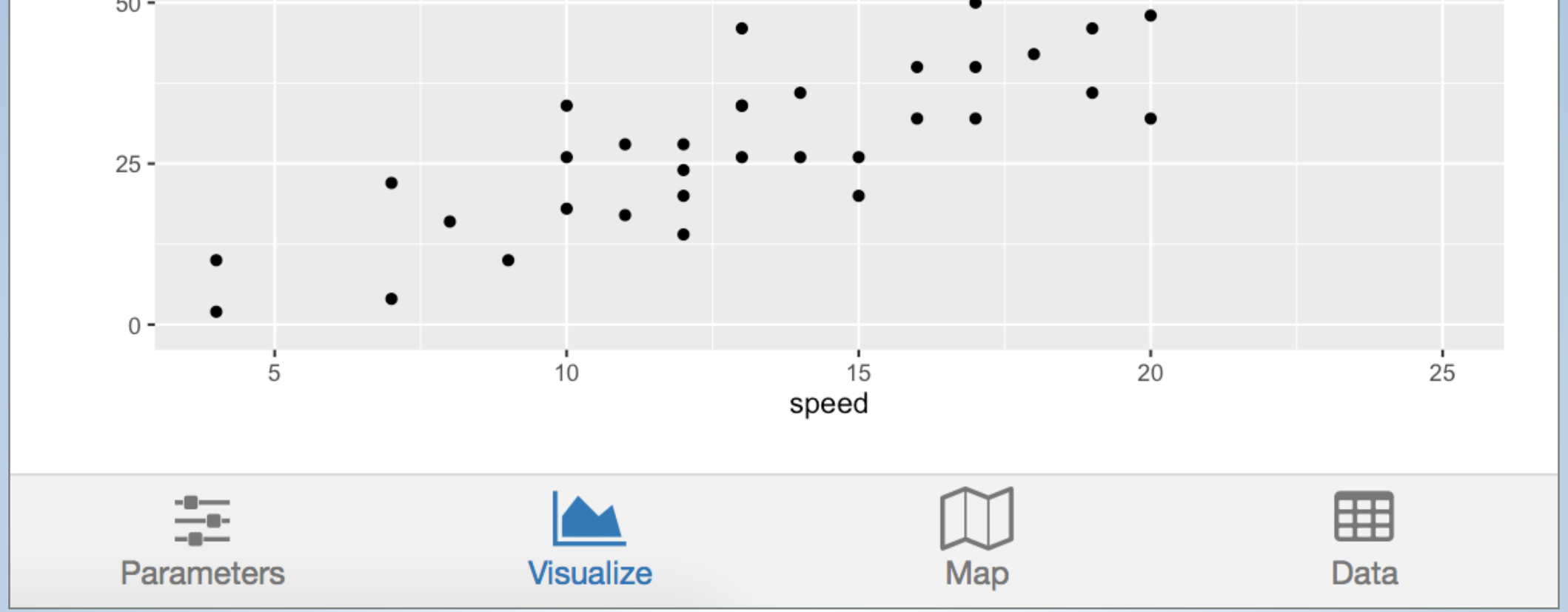
`miniContentPanel()`





button block

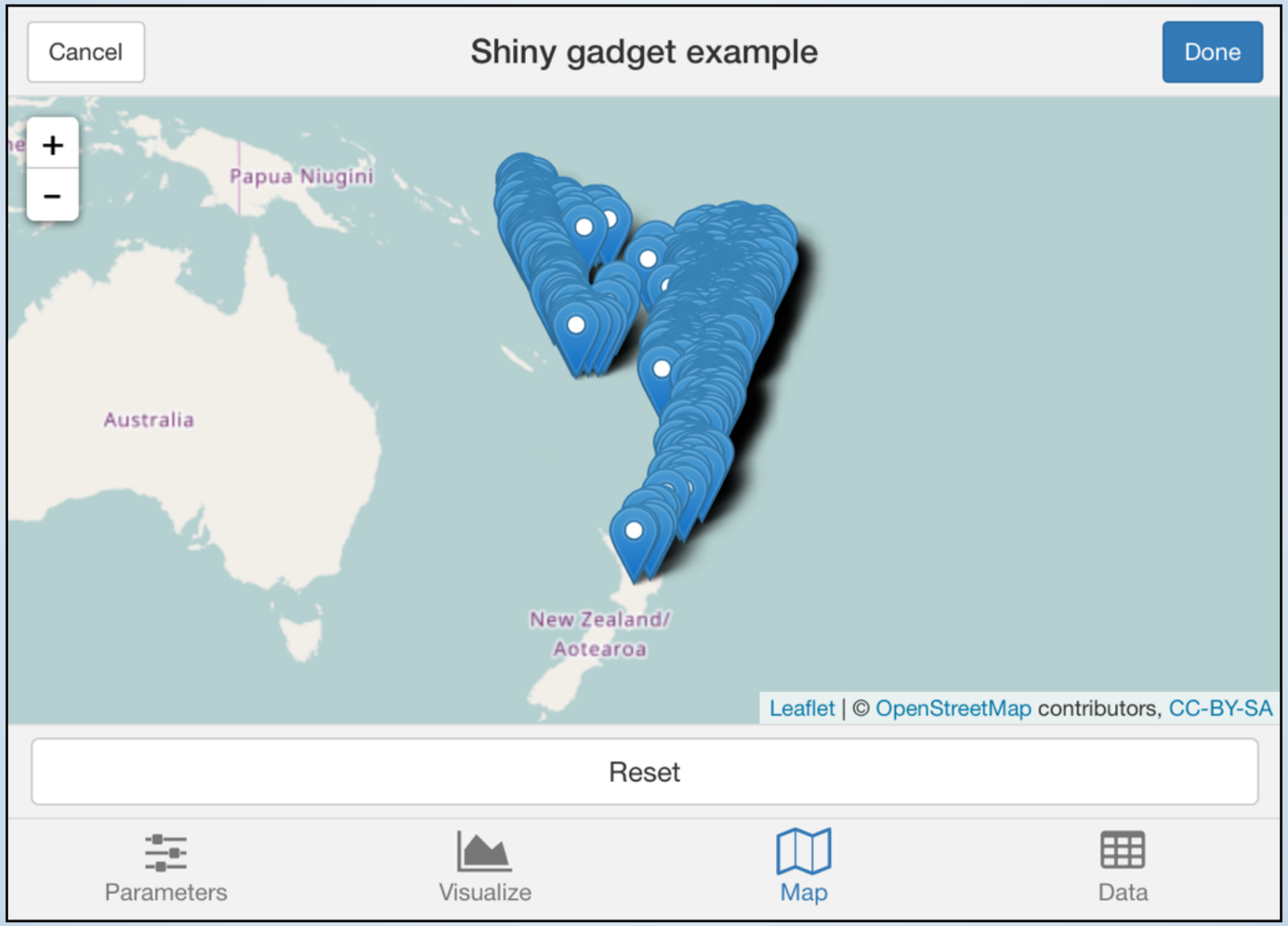
`miniButtonBlock()`



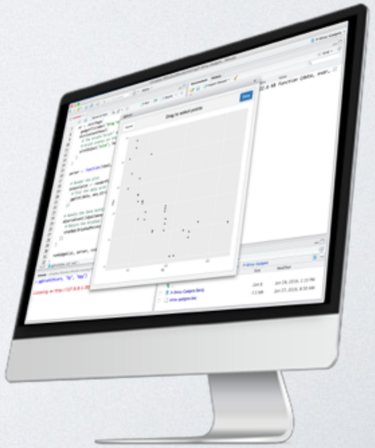
tab strips

`miniTabstripPanel()`

tabs.R

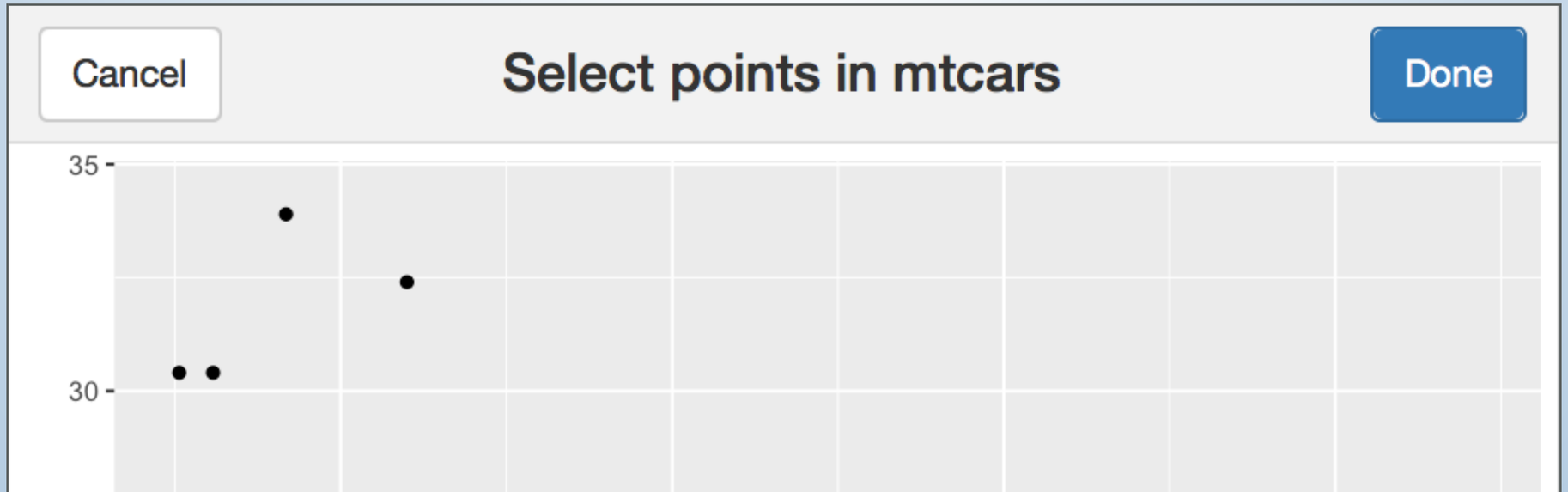

Cancel
Shiny gadget example
Done
Papua Niugini
Australia
New Zealand/
Aotearoa
Leaflet | © OpenStreetMap contributors, CC-BY-SA
Reset
Parameters
Visualize
Map
Data

3. STOPAPP



observeEvent

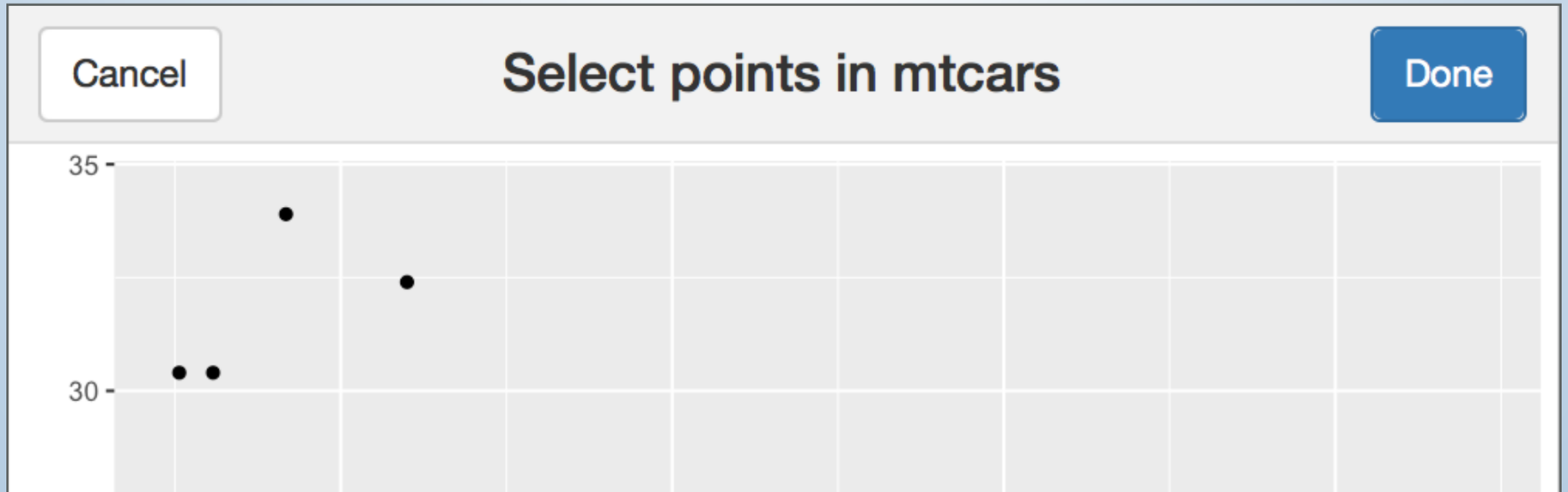
```
observeEvent(input$cancel, { stopApp(NULL) })
observeEvent(input$done, { stopApp(vals$keep) })
```



observeEvent

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

When this changes…

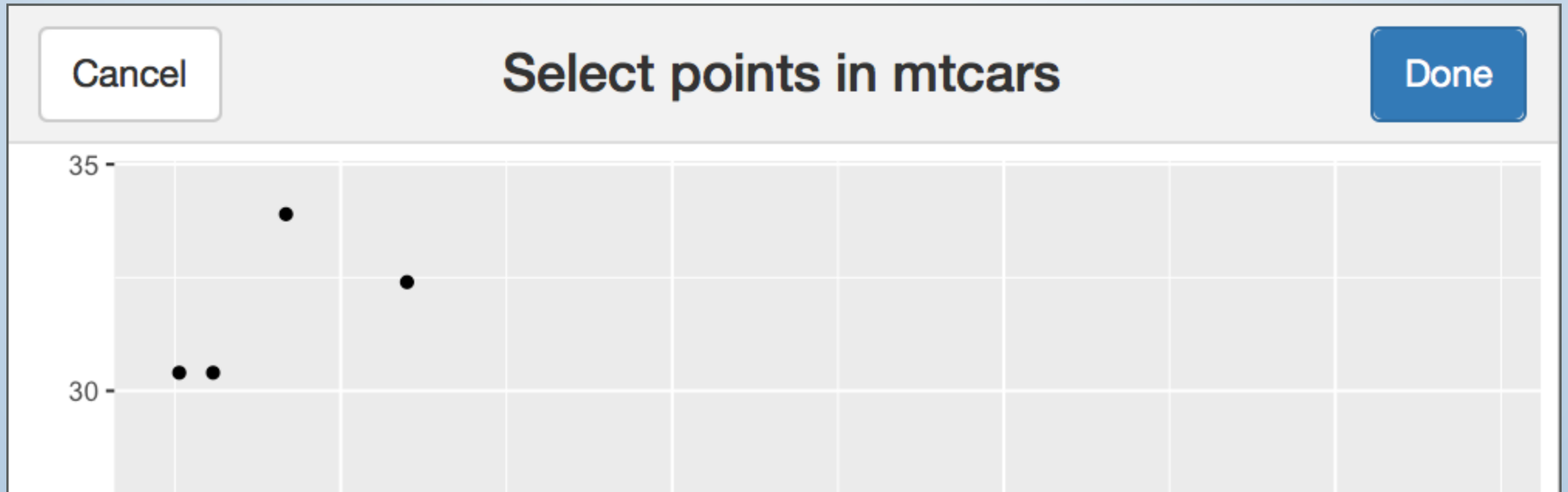


observeEvent

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

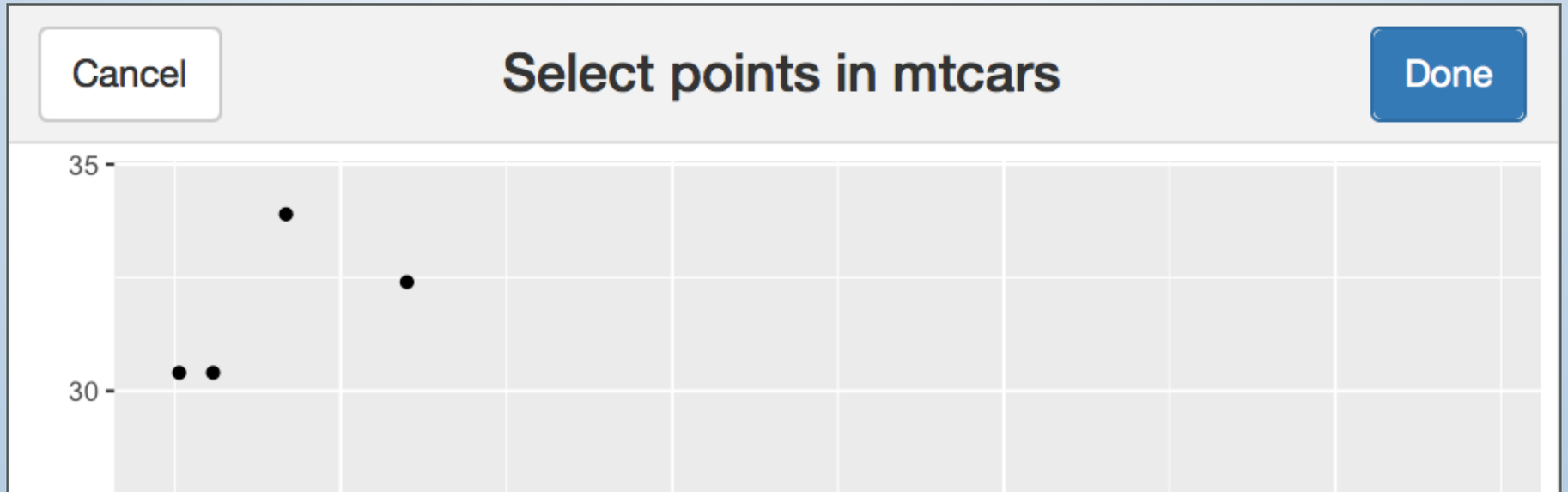
When this changes…

run this.



stopApp

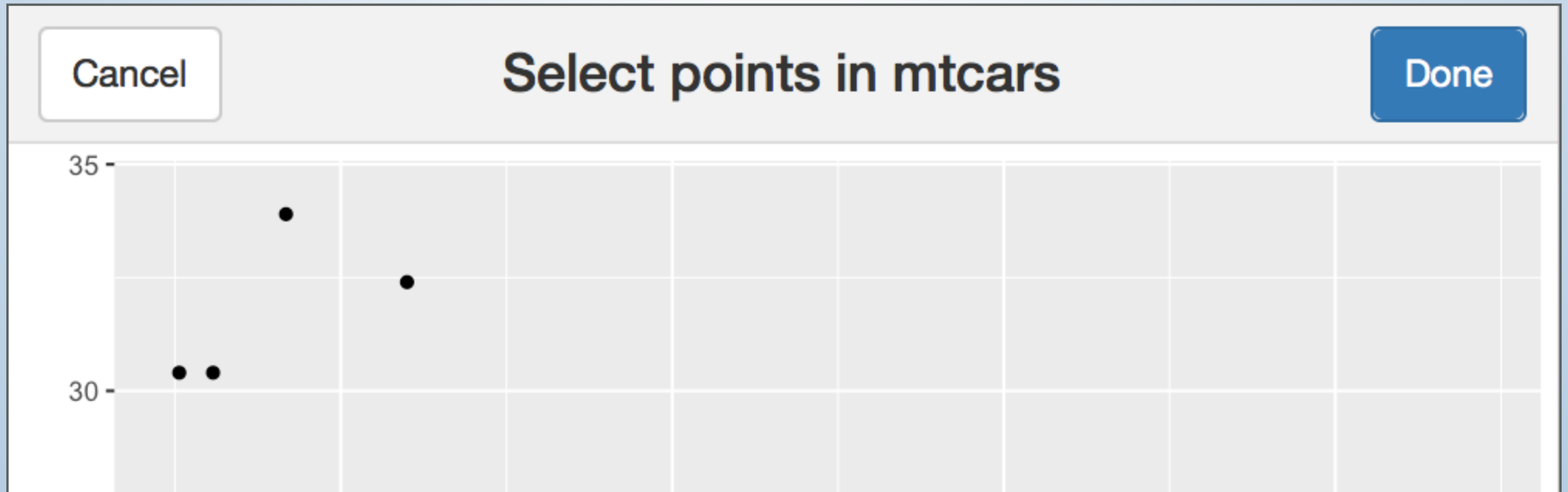
```
observeEvent(input$cancel, { stopApp(NULL) })
observeEvent(input$done, { stopApp(vals$keep) })
```



stopApp

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

Exit the app

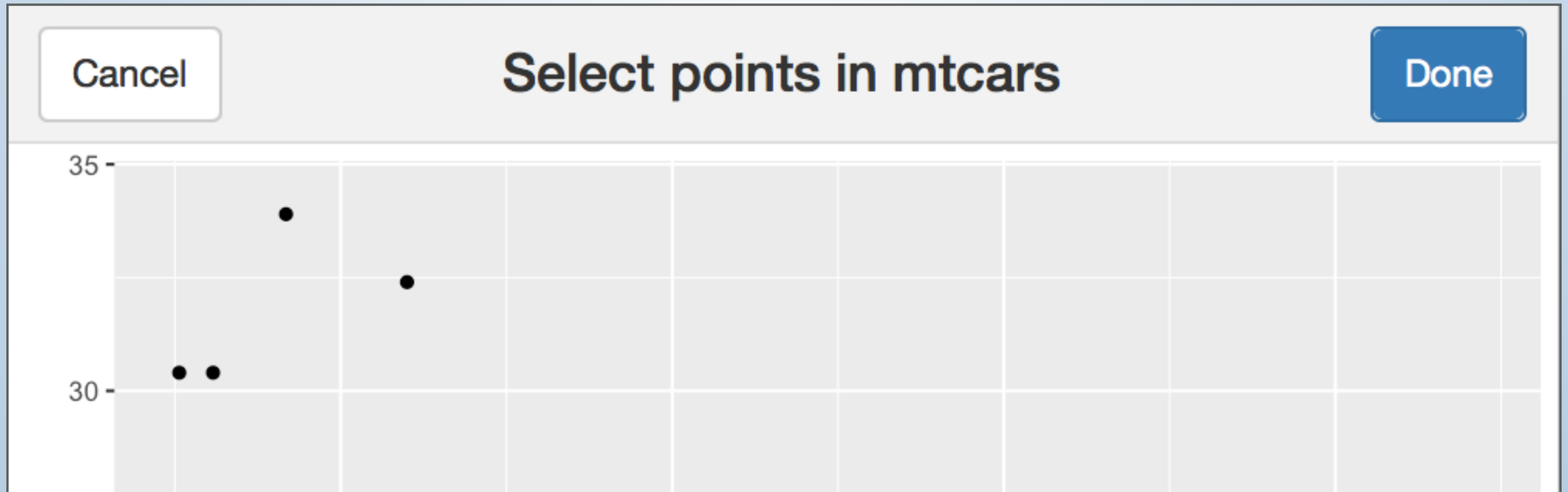


stopApp

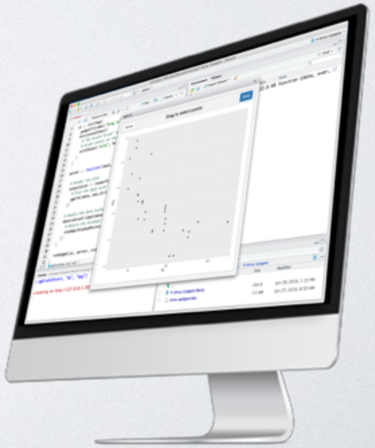
```
observeEvent(input$cancel, { stopApp(NULL) })
observeEvent(input$done, { stopApp(vals$keep) })
```

Exit the app

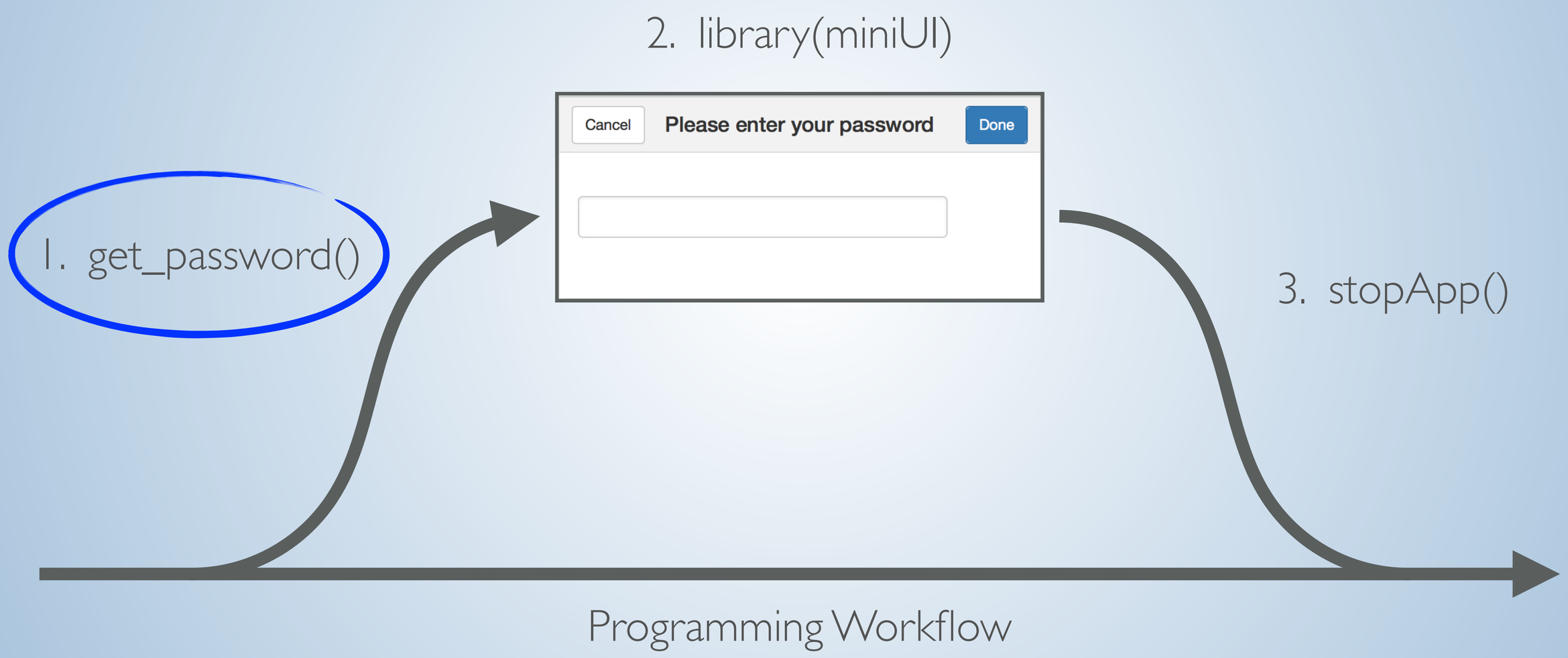
Return this value



ADDINS

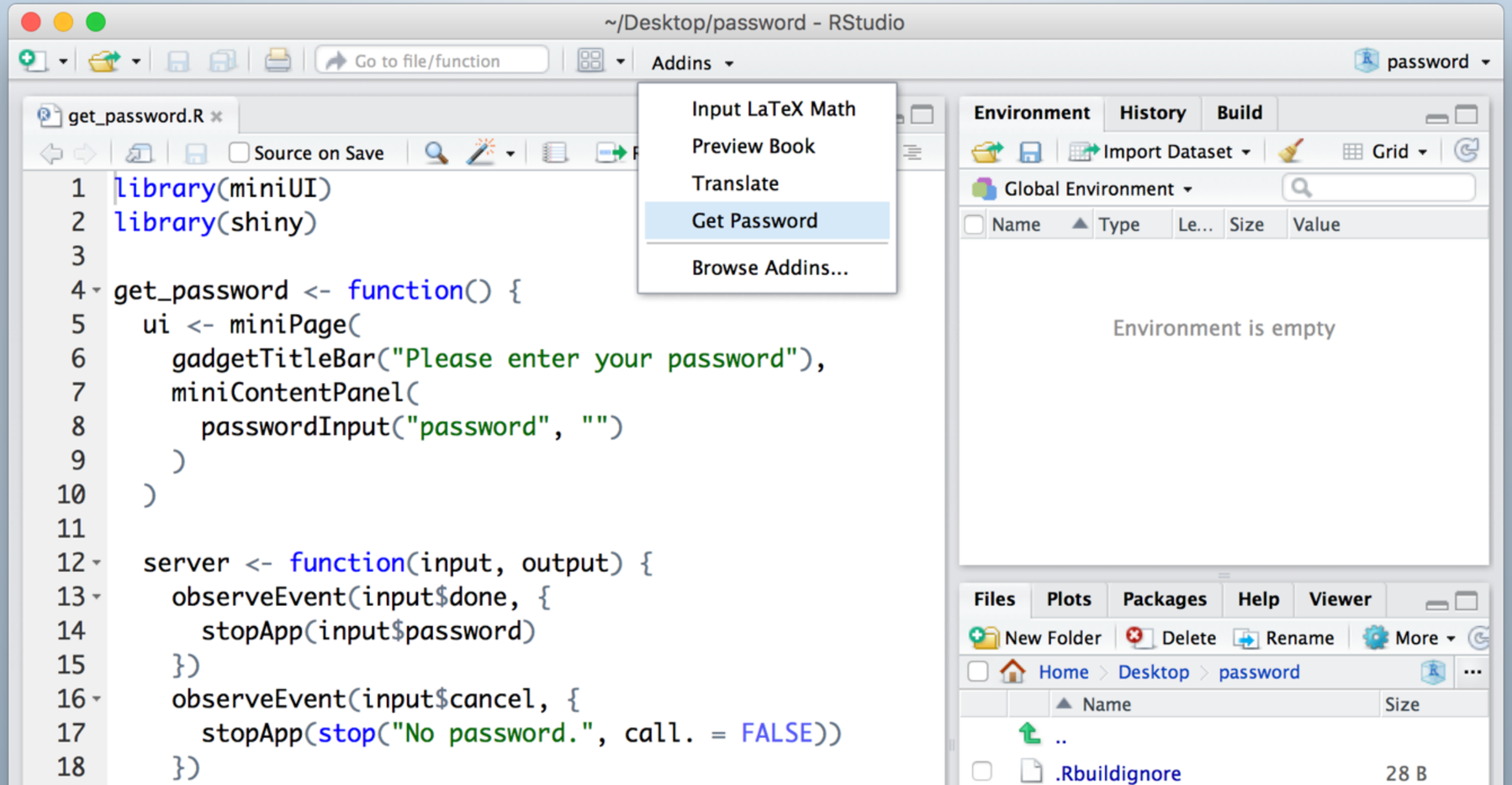


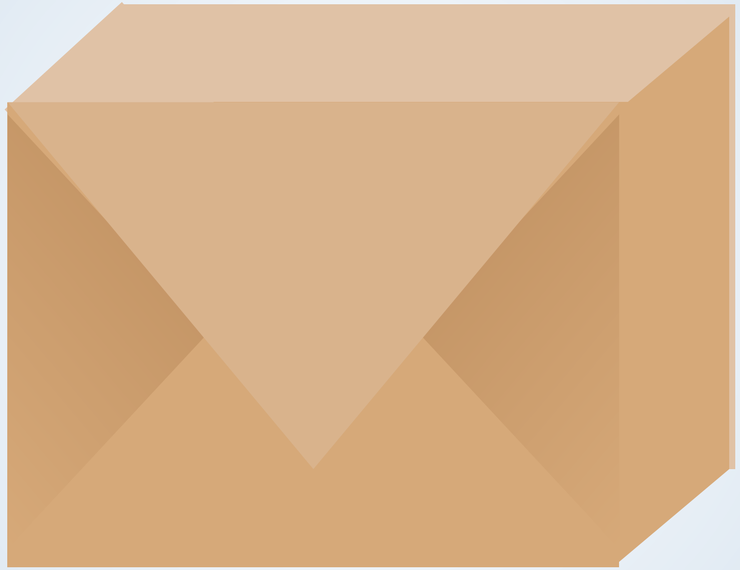
3 Essential Features



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/

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:

THANK YOU

shiny.rstudio.com/articles/gadgets.html
shiny.rstudio.com/articles/gadget-ui.html
rstudio.github.io/rstudioaddins/