615 Shiny

Haoran Cui

2024-11-12

```
library(shiny)
```

#Hadley_1

```
ui <- fluidPage(
  selectInput("dataset", label = "Dataset", choices = ls("package:datasets")),
  verbatimTextOutput("summary"),
  tableOutput("table")
)


server <- function(input, output, session) {
  output$summary <- renderPrint({
    dataset <- get(input$dataset, "package:datasets")
    summary(dataset)
  })

  output$table <- renderTable({
    dataset <- get(input$dataset, "package:datasets")
    dataset
  })
}

shinyApp(ui, server)
```

Dataset

ability.cov

```
       Length Class  Mode
cov    36     -none- numeric
center  6     -none- numeric
n.obs   1     -none- numeric
```

cov.general	cov.picture	cov.blocks	cov.maze	cov.reading	cov.vocab	center	n.c
24.64	5.99	33.52	6.02	20.75	29.70	0.00	112
5.99	6.70	18.14	1.78	4.94	7.20	0.00	112
33.52	18.14	149.83	19.42	31.43	50.75	0.00	112
6.02	1.78	19.42	12.71	4.76	9.07	0.00	112
20.75	4.94	31.43	4.76	52.60	66.76	0.00	112

#Hadley_2

```
ui <- fluidPage(
  selectInput("dataset", label = "Dataset", choices = ls("package:datasets")),
  verbatimTextOutput("summary"),
  tableOutput("table")
)

server <- function(input, output, session) {
  # Create a reactive expression
  dataset <- reactive({
    get(input$dataset, "package:datasets")
  })

  output$summary <- renderPrint({
    # Use a reactive expression by calling it like a function
    summary(dataset())
  })

  output$table <- renderTable({
    dataset()
  })
}
shinyApp(ui, server)
```

Dataset

ability.cov

```
       Length Class  Mode
cov    36     -none- numeric
center  6     -none- numeric
n.obs   1     -none- numeric
```

cov.general	cov.picture	cov.blocks	cov.maze	cov.reading	cov.vocab	center	n.c
24.64	5.99	33.52	6.02	20.75	29.70	0.00	112
5.99	6.70	18.14	1.78	4.94	7.20	0.00	112
33.52	18.14	149.83	19.42	31.43	50.75	0.00	112
6.02	1.78	19.42	12.71	4.76	9.07	0.00	112
20.75	4.94	31.43	4.76	52.60	66.76	0.00	112

2.3.5 Exercises

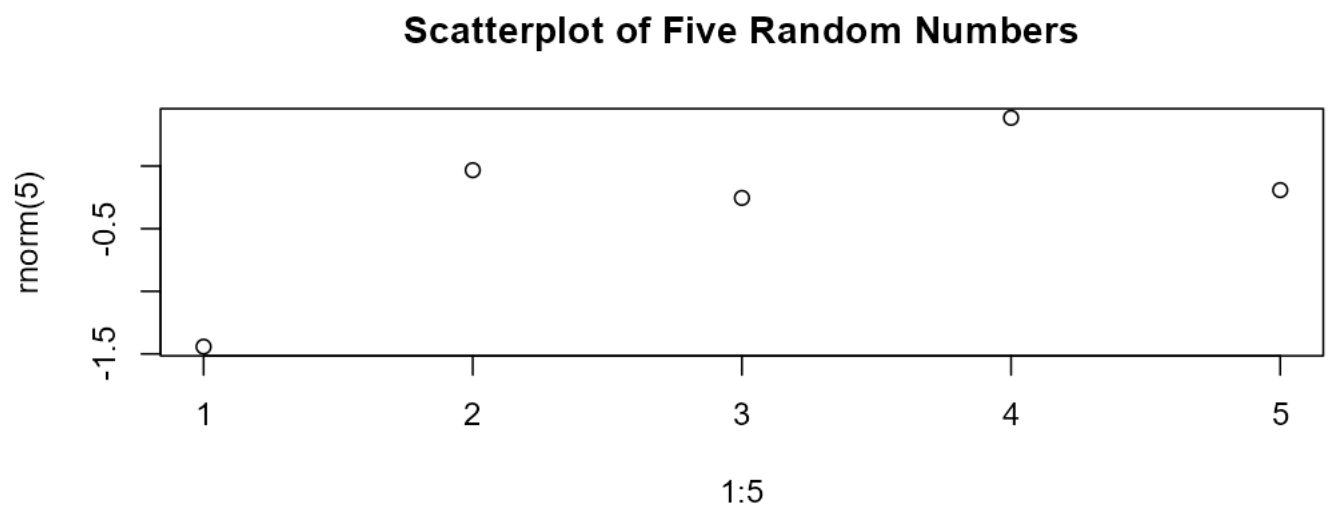
1.

a.verbatimTextOutput() b.textOutput() c.verbatimTextOutput() d.textOutput()

2.

height:300px, width:700px

```
library(shiny)

ui <- fluidPage(
  plotOutput("plot", height = "300px", width = "700px")
)
server <- function(input, output, session) {
  output$plot <- renderPlot({
    plot(1:5, rnorm(5), main = "Scatterplot of Five Random Numbers")
  }, res = 96)
}
#server <- function(input, output, session) {
 # output$plot <- renderPlot(plot(1:5), res = 96)
#}
shinyApp(ui, server)
```



3.

```
ui <- fluidPage(
  dataTableOutput("table")
)
```

```
## `shiny::dataTableOutput()` is deprecated as of shiny 1.8.1.
## Please use `DT::DTOutput()` instead.
## Since you have a suitable version of DT (>= v0.32.1), shiny::dataTableOutput() will automatic
ally use DT::DTOutput() under-the-hood.
## If this happens to break your app, set `options(shiny.legacy.datatable = TRUE)` to get the le
gacy datatable implementation (or `FALSE` to squelch this message).
## See <https://rstudio.github.io/DT/shiny.html> for more information.
```

```
server <- function(input, output, session) {
  output$table <- DT::renderDT(mtcars, options = list(pageLength = 5))
}

shinyApp(ui, server)
```

Show 5 entries Search:

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear
Mazda RX4	21	6	160	110	3.9	2.62	16.46	0	1	4
Mazda RX4 Wag	21	6	160	110	3.9	2.875	17.02	0	1	4
Datsun 710	22.8	4	108	93	3.85	2.32	18.61	1	1	4
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3
Hornet Sportabout	18.7	8	360	175	3.15	3.44	17.02	0	0	3

4.

```
library(reactable)

ui <- fluidPage(
  reactableOutput("table")
)

server <- function(input, output, session) {
  output$table <- renderReactable({
    reactable(
      mtcars,
      pagination = TRUE,          # Enable pagination
      searchable = FALSE,         # Disable the search box
      sortable = FALSE,           # Disable column sorting
      showPageSizeOptions = FALSE # Disable option to change number of rows displayed
    )
  })
}

shinyApp(ui, server)
```

	mpg	cyl	disp	hp	drat
Mazda RX4	21	6	160	110	3.9
Mazda RX4 Wag	21	6	160	110	3.9
Datsun 710	22.8	4	108	93	3.85
Hornet 4 Drive	21.4	6	258	110	3.08
Hornet Sportabout	18.7	8	360	175	3.15
Valiant	18.1	6	225	105	2.76
Duster 360	14.3	8	360	245	3.21
Merc 240D	24.4	4	146.7	62	3.69
Merc 230	22.8	4	140.8	95	3.92

3.3.6 Exercises

1.

```
ui <- fluidPage(
  textInput("name", "What's your name?"),
  textOutput("greeting")
)
```

server1

```
server1 <- function(input, output, server) {
  output$greeting <- renderText({
    paste0("Hello ", input$name)
  })
}
```

server2

```
server2 <- function(input, output, server) {
  output$greeting <- renderText({
    paste0("Hello ", input$name)
  })
}
```

server3

```
server3 <- function(input, output, server) {
  output$greeting <- renderText({
    paste0("Hello ", input$name)
  })
}
```

2.

server1

```
server1 <- function(input, output, session) {
  c <- reactive(input$a + input$b)
  e <- reactive(c() + input$d)
  output$f <- renderText(e())
}
```

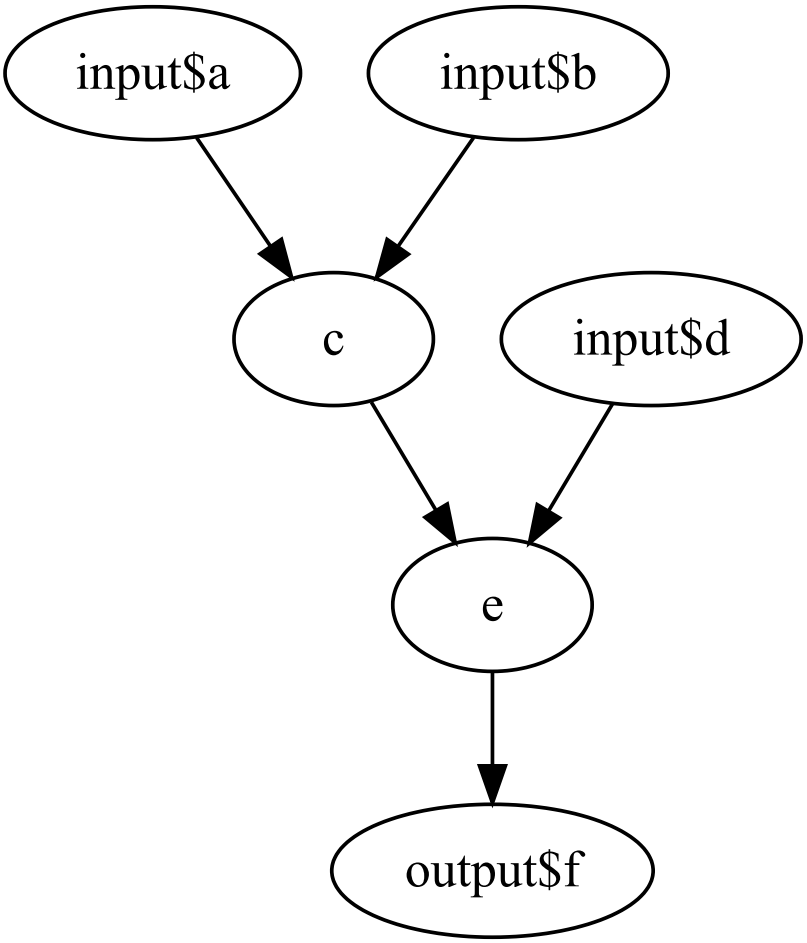
server2

```
server2 <- function(input, output, session) {
  x <- reactive(input$x1 + input$x2 + input$x3)
  y <- reactive(input$y1 + input$y2)
  output$z <- renderText(x() / y())
}
```

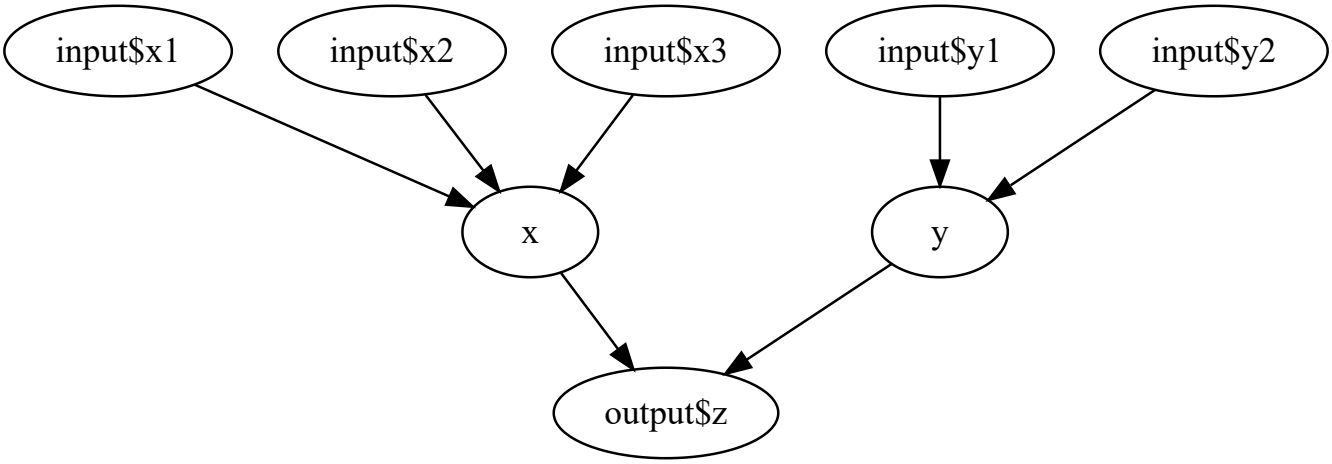
server3

```
server3 <- function(input, output, session) {
  d <- reactive(c() ^ input$d)
  a <- reactive(input$a * 10)
  c <- reactive(b() / input$c)
  b <- reactive(a() + input$b)
}
```

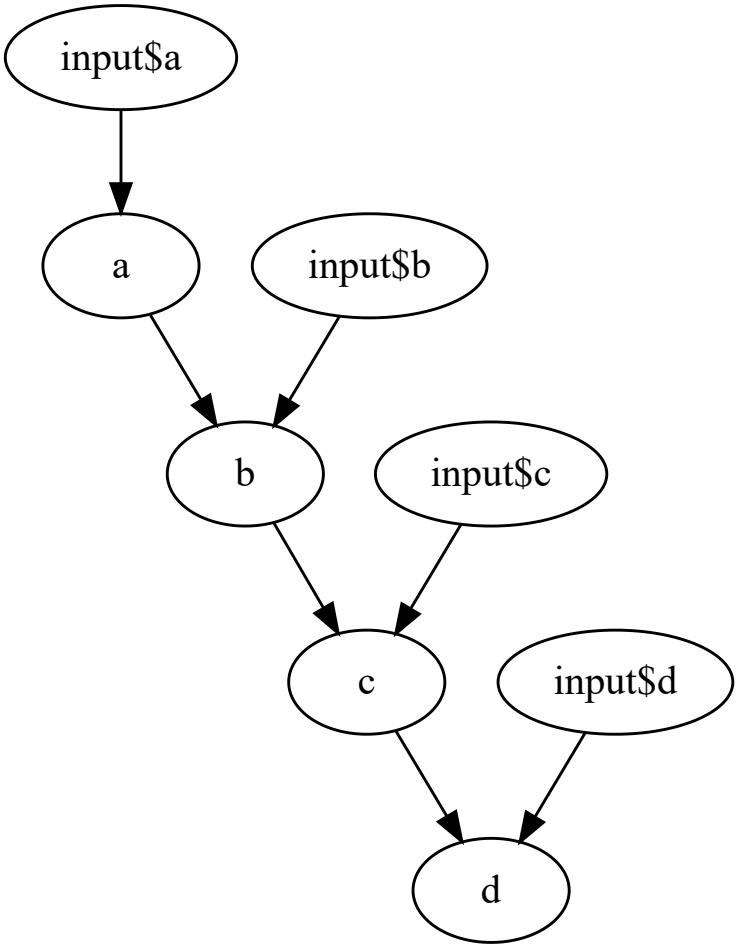
```r
# Load necessary packages
library(DiagrammeR)

# Define the graphs for each server function

# Graph for server1
graph_server1 <- grViz("
digraph server1 {
  'input$a' -> 'c'
  'input$b' -> 'c'
  'c' -> 'e'
  'input$d' -> 'e'
  'e' -> 'output$f'
}
")

# Graph for server2
graph_server2 <- grViz("
digraph server2 {
  'input$x1' -> 'x'
  'input$x2' -> 'x'
  'input$x3' -> 'x'
  'x' -> 'output$z'
  'input$y1' -> 'y'
  'input$y2' -> 'y'
  'y' -> 'output$z'
}
")

# Graph for server3
graph_server3 <- grViz("
digraph server3 {
  'input$a' -> 'a'
  'a' -> 'b'
  'input$b' -> 'b'
  'b' -> 'c'
  'input$c' -> 'c'
  'c' -> 'd'
  'input$d' -> 'd'
}
")

# Display the graphs
graph_server1
```




```
graph_server2
```




```
graph_server3
```



3.

```
selected_var <- reactive(df[[input$var]])
var_range <- reactive(range(selected_var(), na.rm = TRUE))
```

The code fails because “var” and “range” are names of existing R functions. var() for variance and range() for the range of a vector. Using them as reactive names can cause conflicts and make the code unclear.

4.8 Exercises

1.

```
                ┌──────────────┐
                │ input$code   │
                └──────┬───────┘
                       │
            ┌──────────▼──────────┐
            │ selected()          │
            │ (filters injuries   │
            │  based on prod_code)│
            └──┬──────────┬───────┘
               │          │
   ┌───────────▼──┐  ┌────▼──────────┐
   │ output$diag  │  │ output$body_part│
   │ (table count │  │ (table count   │
   │  by diag)    │  │  by body_part) │
   └──────────────┘  └───────────────┘
                 │
                 │
              ┌──▼──────────────┐
              │ output$location │
              │ (table count    │
              │  by location)   │
              └──▲──────────────┘
                 │
          ┌──────┴──────────────┐
          │ summary()           │
          │ (count by age, sex, │
          │  then joined with   │
          │  population)        │
          └──────┬──────────────┘
                 │
        ┌────────▼──────────┐
        │ output$age_sex    │
        │ (line plot by age,│
        │  n per age-sex group)│
        └───────────────────┘
```

```
                ┌──────────────┐
                │ input$code   │
                └──────┬───────┘
                       │
            ┌──────────▼──────────┐
            │ selected()          │
            │ (filters injuries   │
            │  based on prod_code)│
            └──┬──────────┬───────┘
               │          │
```

```
┌──────────────▼──┐ ┌──▼──────────────┐
│ output$diag     │ │ output$body_part│
│ (table count    │ │ (table count    │
│  by diag)       │ │  by body_part)  │
└─────────────────┘ └─────────────────┘
                │
                │
            ┌───▼──────────────┐
            │ output$location  │
            │ (table count     │
            │  by location)    │
            └──▲───────────────┘
                │
        ┌───────┴──────────────┐
        │ summary()            │
        │ (count by age, sex,  │
        │  then joined with    │
        │  population)         │
        └──────────────────────┘
                │
                │
   ┌────────────▼──────────────────────────┐
   │ input$y                               │
   │ (user choice: rate vs count)          │
   └───────────────────────────────────────┘
                │
                │
      ┌─────────▼────────────┐
      │ output$age_sex       │
      │ (conditional plot by │
      │  age: rate or count  │
      │  based on input$y)   │
      └──────────────────────┘
```

```
            ┌───────────────┐
            │ input$code    │
            └───────────────┘
                    │
                    │
          ┌─────────▼──────────┐
          │ selected()         │
          │ (filters injuries  │
          │  based on prod_code)│
          └────────────────────┘
              │          │
              │          │
┌──────────────▼──┐ ┌──▼──────────────┐
│ output$diag     │ │ output$body_part│
│ (table count    │ │ (table count    │
│  by diag)       │ │  by body_part)  │
└─────────────────┘ └─────────────────┘
                │
                │
```

```
              ┌──▼──────────────┐
              │ output$location │
              │ (table count    │
              │  by location)   │
              └──▲──────────────┘
                 │
            ┌────┴────────────────┐
            │ summary()           │
            │ (count by age, sex, │
            │  then joined with   │
            │  population)        │
            └─────────────────────┘
                      │
                      │
           ┌──────────▼──────────┐
           │ output$age_sex      │
           │ (conditional plot by │
           │  age: rate or count) │
           └─────────────────────┘
                      │
                      │
           ┌──────────▼──────────┐
           │ input$story         │
           └─────────────────────┘
                      │
                      │
          ┌───────────▼──────────┐
          │ narrative_sample()   │
          │ (gets new narrative  │
          │  on button click)    │
          └──────────────────────┘
                      │
                      │
            ┌─────────▼──────────┐
            │ output$narrative   │
            │ (displays current  │
            │  narrative text)   │
            └────────────────────┘
```

2.

If you reverse the order of fct_infreq() and fct_lump(), the code will first lump all values and then order them by frequency. This approach may result in a less accurate summary table, as less common factors could be grouped together with more common ones. This can reduce the interpretability and accuracy of the summarized data.

3.

Add a slider input for selecting the number of rows to display column(4, sliderInput("num_rows", label = "Number of rows:", min = 1, max = 10, value = 5))

Render tables for diagnosis, body part, and location, showing the top rows based on the selected number output\(diag <- renderTable({ count_top(selected(), diag, n = input\)num_rows) }, width = "100%")

output\(body_part <- renderTable({ count_top(selected(), body_part, n = input\)num_rows) }, width = "100%")

output\(location <- renderTable({ count_top(selected(), location, n = input\)num_rows) }, width = "100%")

4.

```
fluidRow(
  column(1, actionButton("prev_story", "Previous")),
  column(1, actionButton("next_story", "Next")),
  column(10, textOutput("narrative"))
)
```

Previous

Next

Reactive value to track the current narrative index narrative_index <- reactiveVal(1)

Update the narrative index when the “Next Story” button is clicked observeEvent(input$next_story, { current <- narrative_index() narrative_index(min(current + 1, nrow(selected()))) })

Update the narrative index when the “Previous Story” button is clicked observeEvent(input$prev_story, { current <- narrative_index() narrative_index(max(current - 1, 1)) })

Render the current narrative text based on the narrative index output$narrative <- renderText({ selected() %>% pull(narrative) %>% .[narrative_index()] })