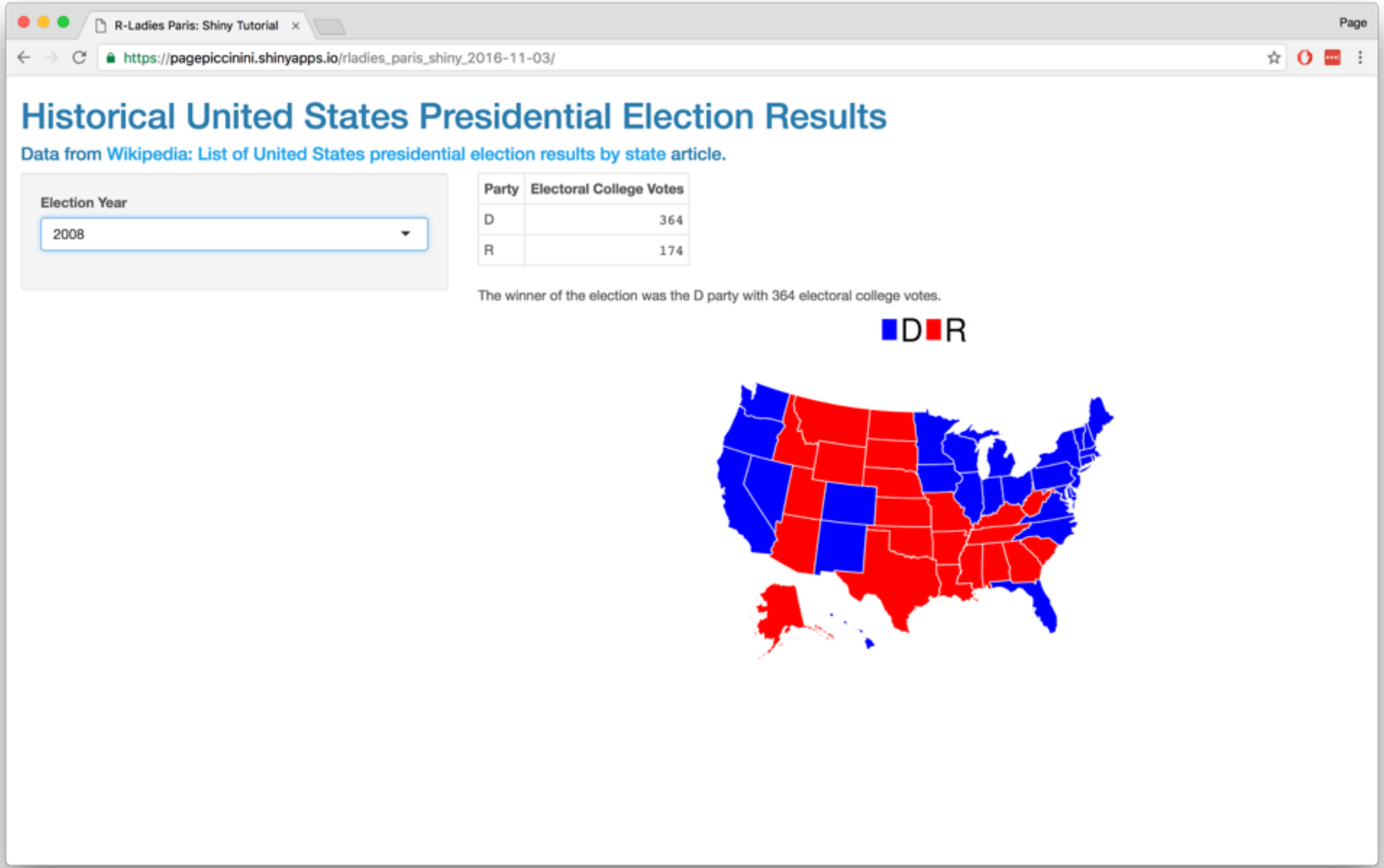
Introduction to Interactive Visualizations with Shiny

Page Piccinini

What is Shiny?


R-Ladies Paris: Shiny Tutorial
https://pagepiccinini.shinyapps.io/rladies_paris_shiny_2016-11-03/
Page

Historical United States Presidential Election Results

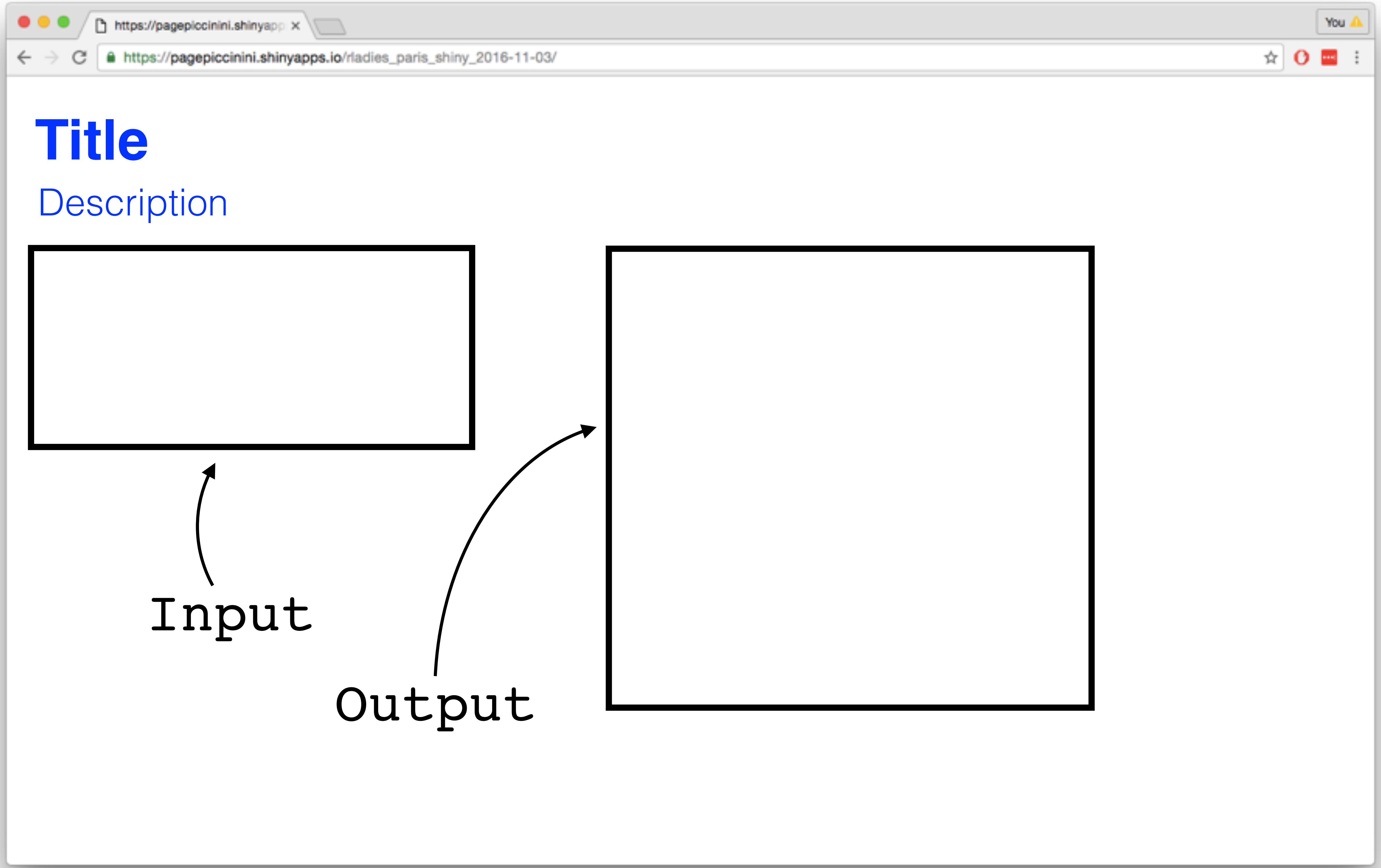
Data from Wikipedia: List of United States presidential election results by state article.

Election Year: 2008

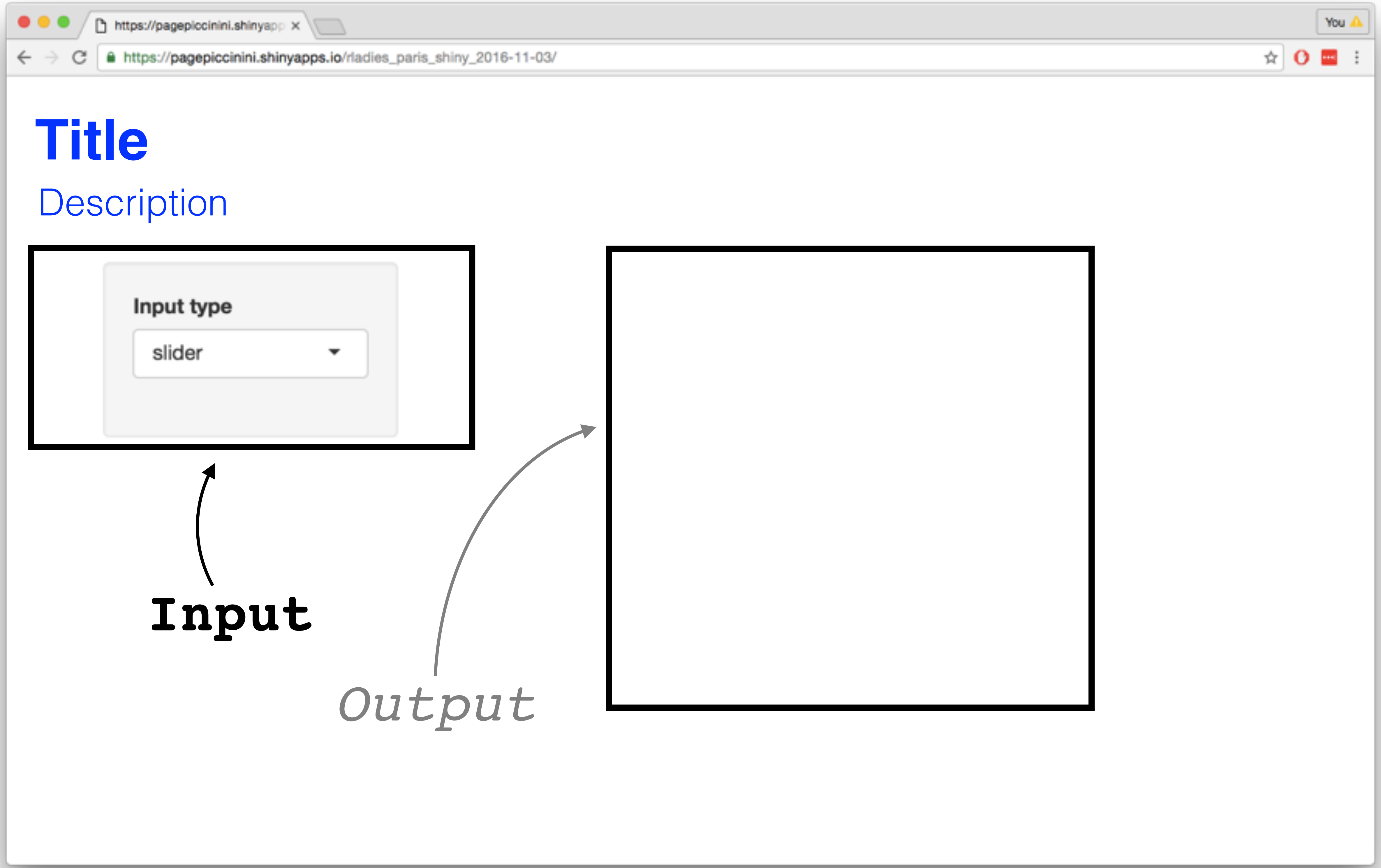
Party	Electoral College Votes
D	364
R	174

The winner of the election was the D party with 364 electoral college votes.

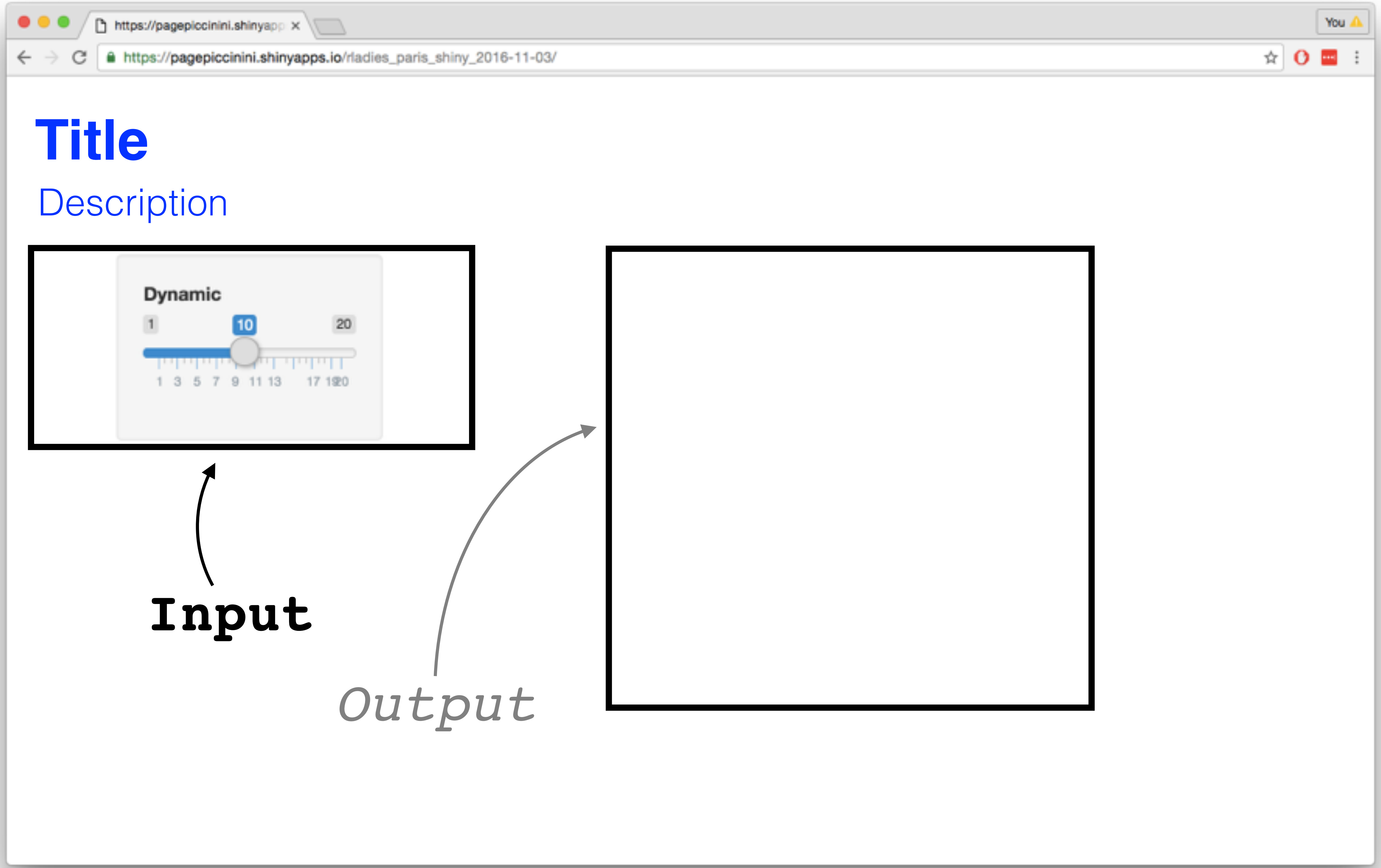
`ui`
(front end)


https://pagepiccinini.shinyapp
https://pagepiccinini.shinyapps.io/rladies_paris_shiny_2016-11-03/
You
Title
Description
Input
Output

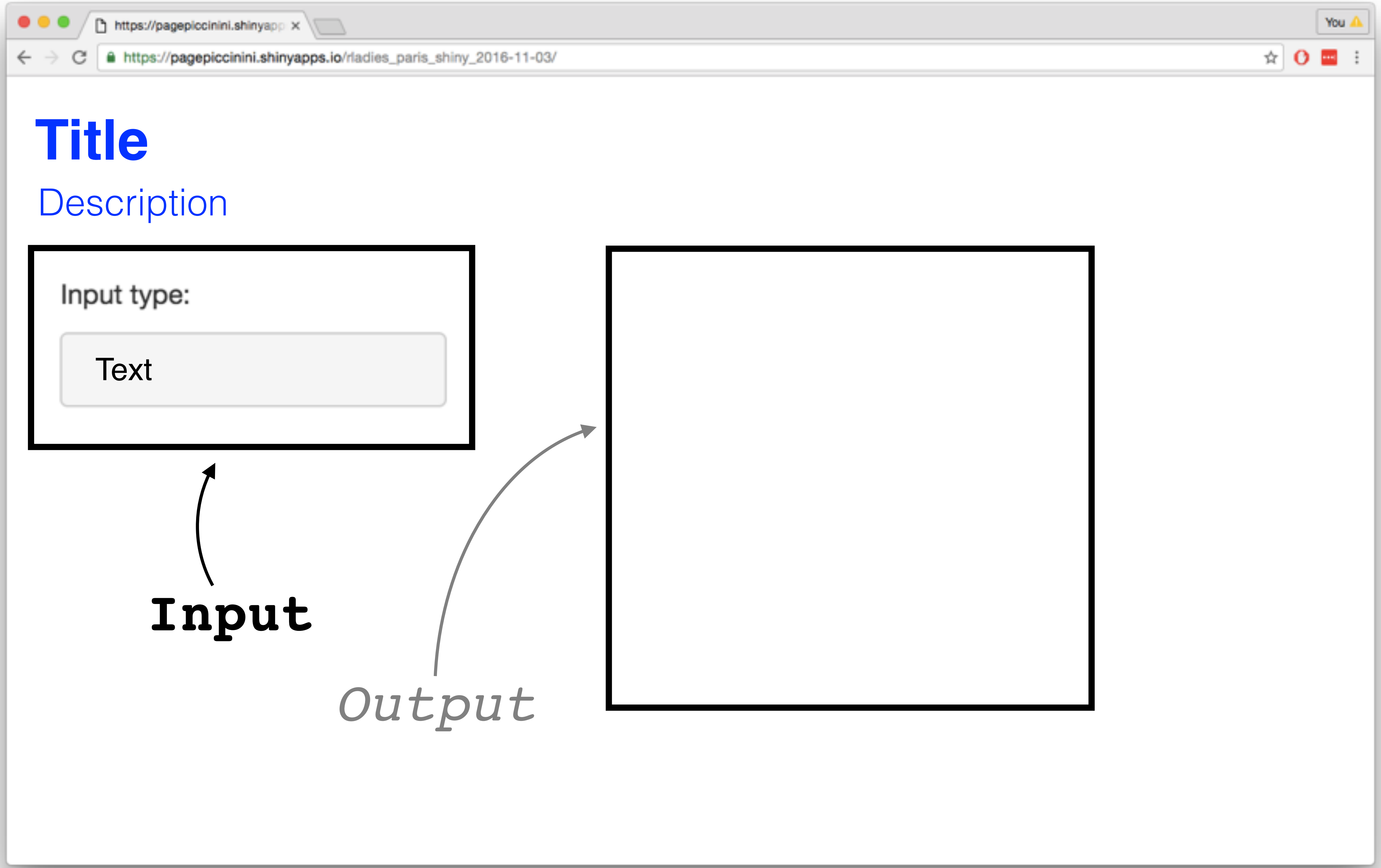
`ui`
(front end)


https://pagepiccinini.shinyapp
https://pagepiccinini.shinyapps.io/rladies_paris_shiny_2016-11-03/
You
Title
Description
Input type
slider
Input
Output

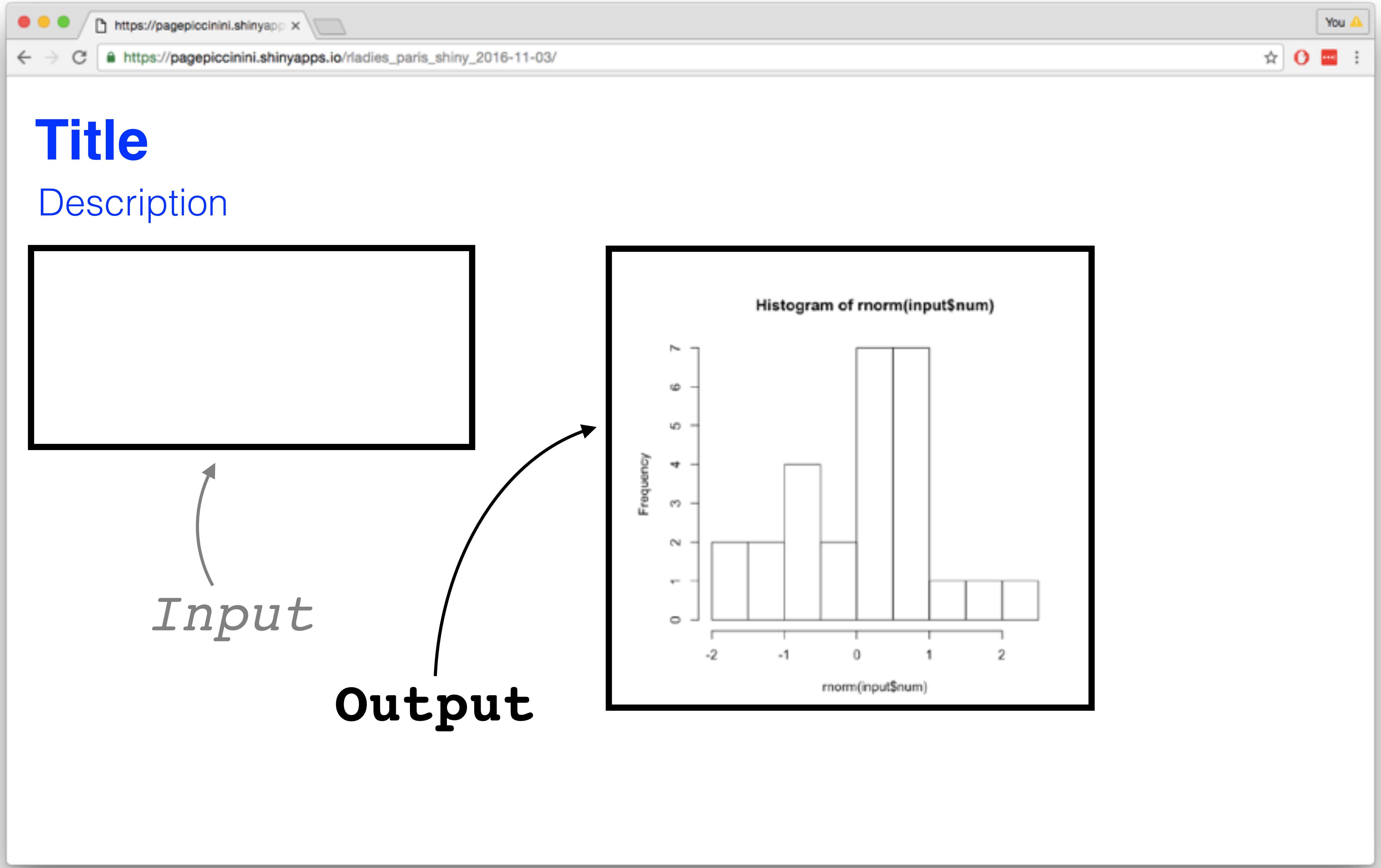
`ui`
(front end)


https://pagepiccinini.shinyapps.io/rladies_paris_shiny_2016-11-03/
Title
Description
Dynamic
1
10
20
1 3 5 7 9 11 13 17 19 20
Input
Output

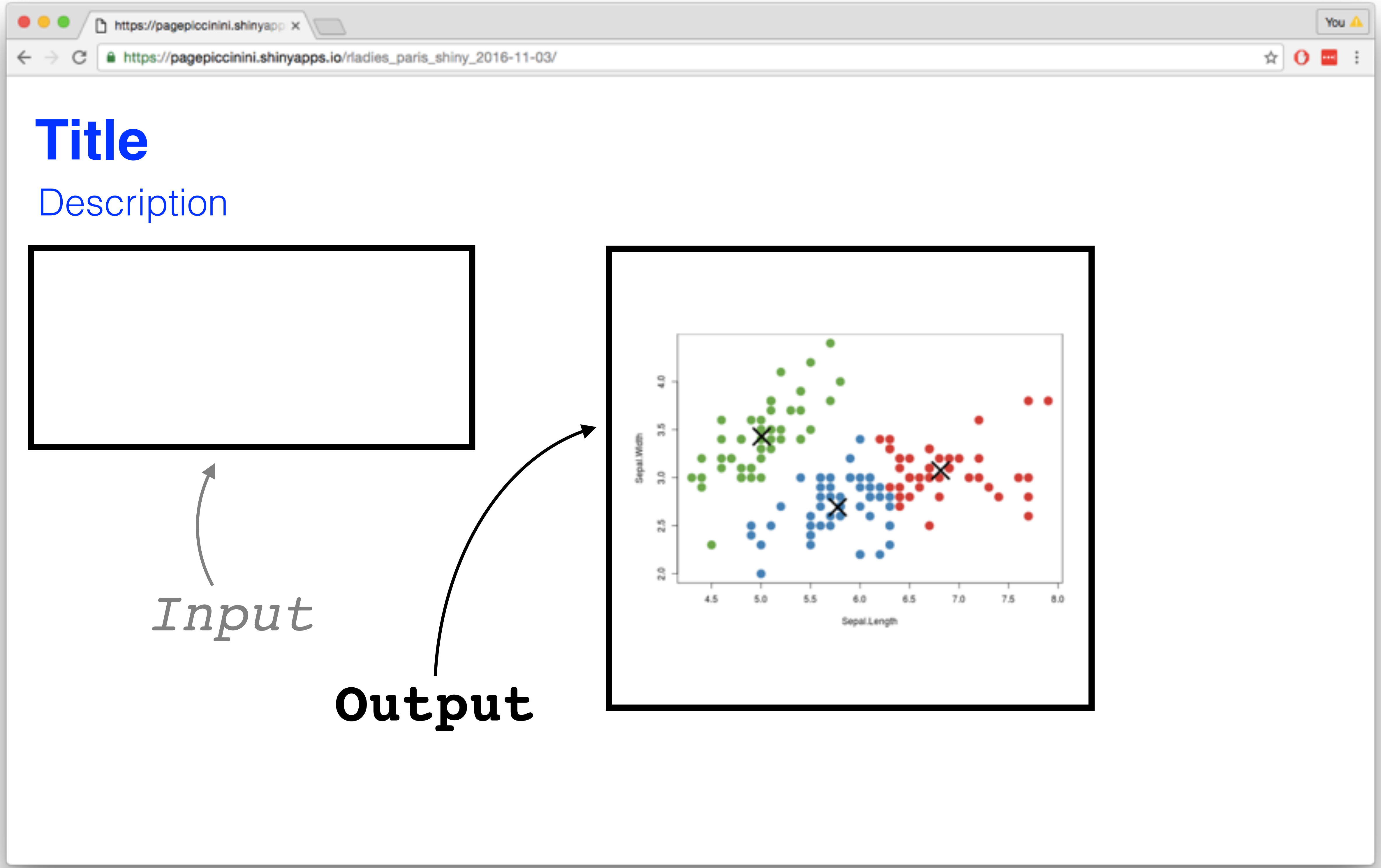
`ui`
(front end)


https://pagepiccinini.shinyapp
https://pagepiccinini.shinyapps.io/rladies_paris_shiny_2016-11-03/
You
Title
Description
Input type:
Text
Input
Output

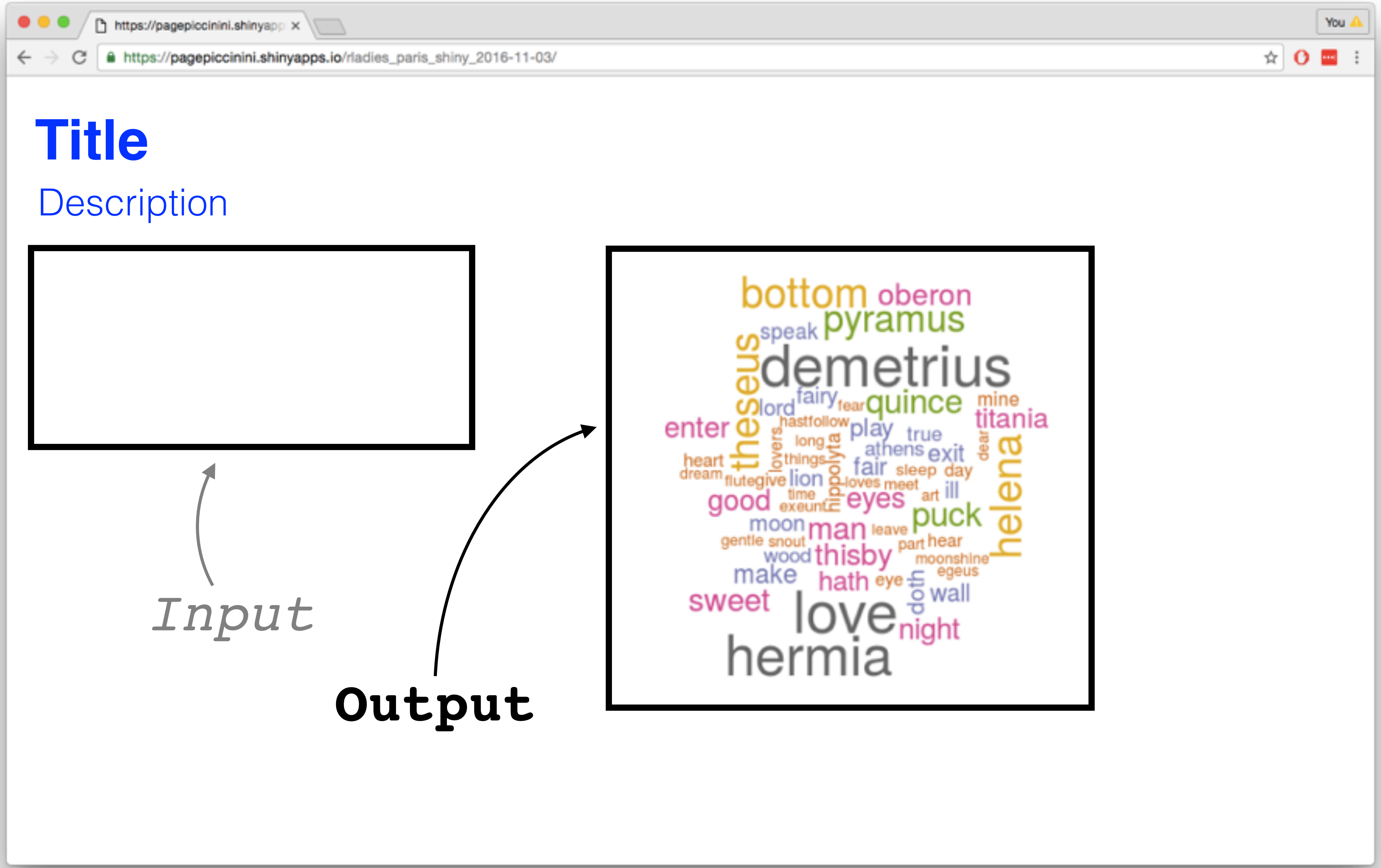
`ui`
(front end)


https://pagepiccinini.shinyapps.io/rladies_paris_shiny_2016-11-03/
Title
Description
Input
Output
Histogram of rnorm(input$num)
Frequency
rnorm(input$num)
-2
-1
0
1
2
0
1
2
3
4
5
6
7

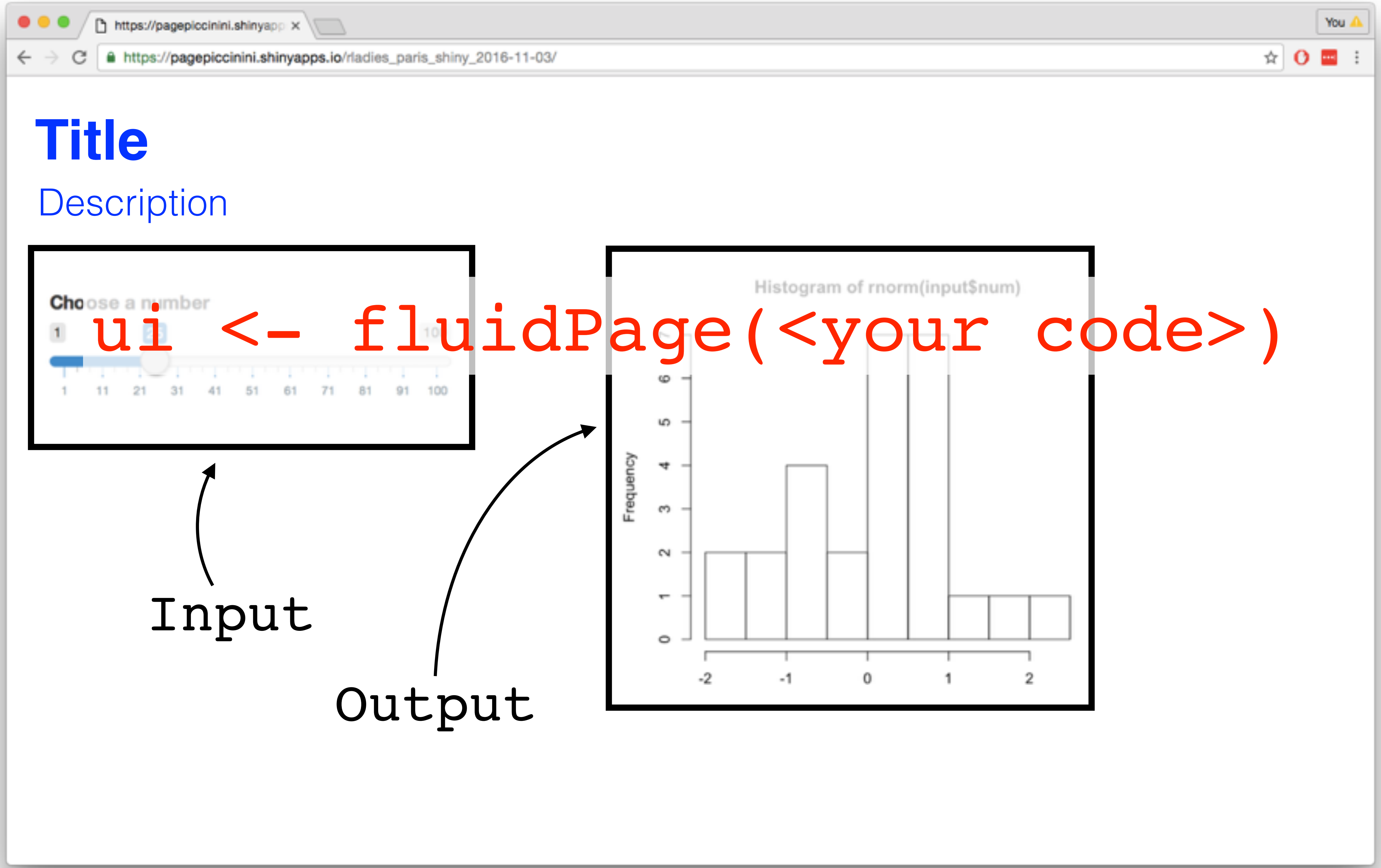
`ui`
(front end)


https://pagepiccinini.shinyapp
https://pagepiccinini.shinyapps.io/rladies_paris_shiny_2016-11-03/
You
Title
Description
Input
Output
Sepal.Width
Sepal.Length

`ui`
(front end)

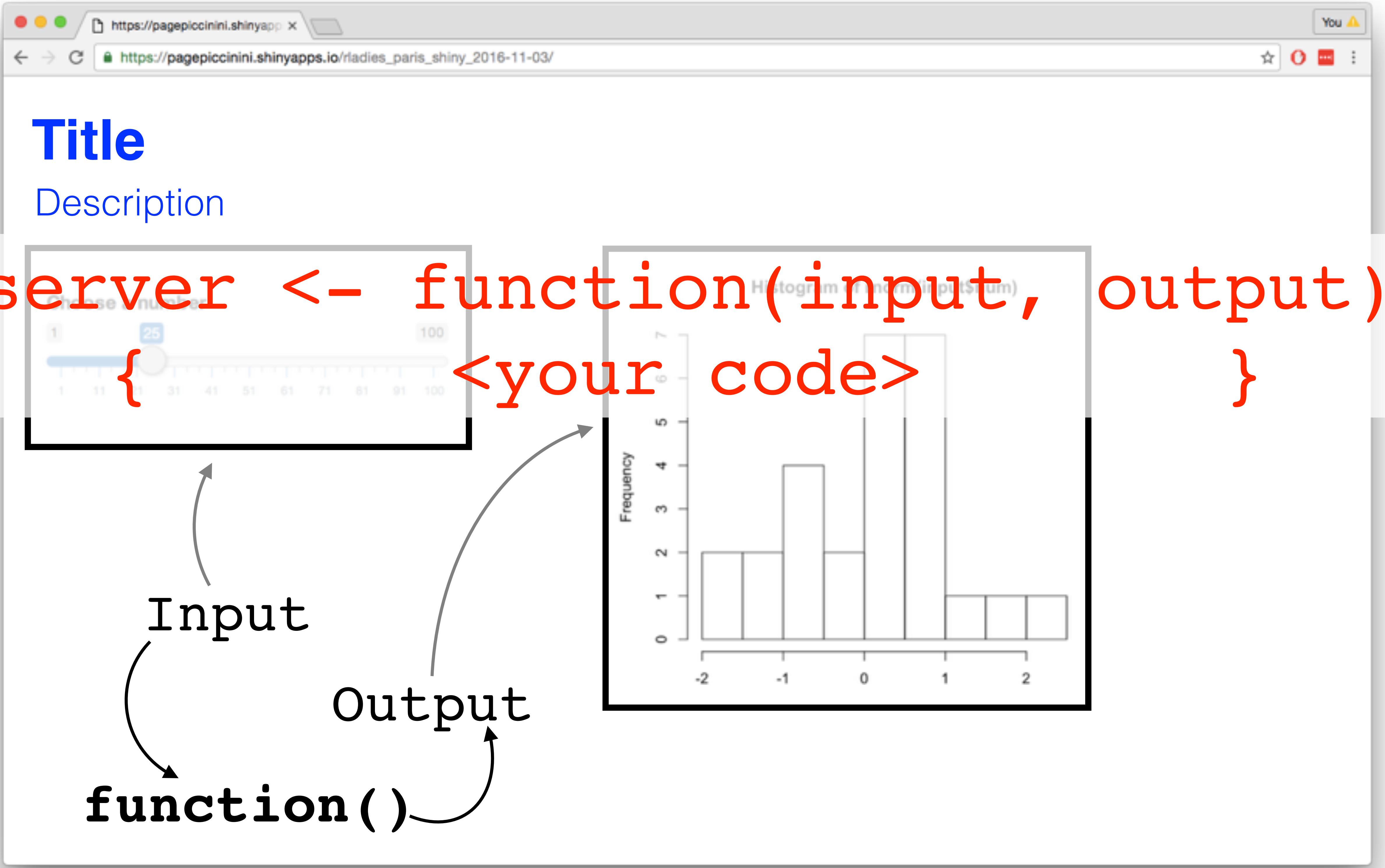

https://pagepiccinini.shinyapps.io/rladies_paris_shiny_2016-11-03/
Title
Description
Input
Output
bottom oberon pyramus speak theseus demetrius fairy fear quince mine titania enter lord hastfollow play true long athens exit heart dream lovers things hippolyta fair sleep day dear flutegive lion loves meet helena good time exeunt eyes art ill moon man leave puck gentle snout part hear wood thisby moonshine make egeus hath eye doth wall sweet love night hermia

`ui`
(front end)



```
ui <- fluidPage(<your code>)
```

server
(back end)


https://pagepiccinini.shinyapp
https://pagepiccinini.shinyapps.io/rladies_paris_shiny_2016-11-03/
Title
Description
Input
Output
function()

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

app.R

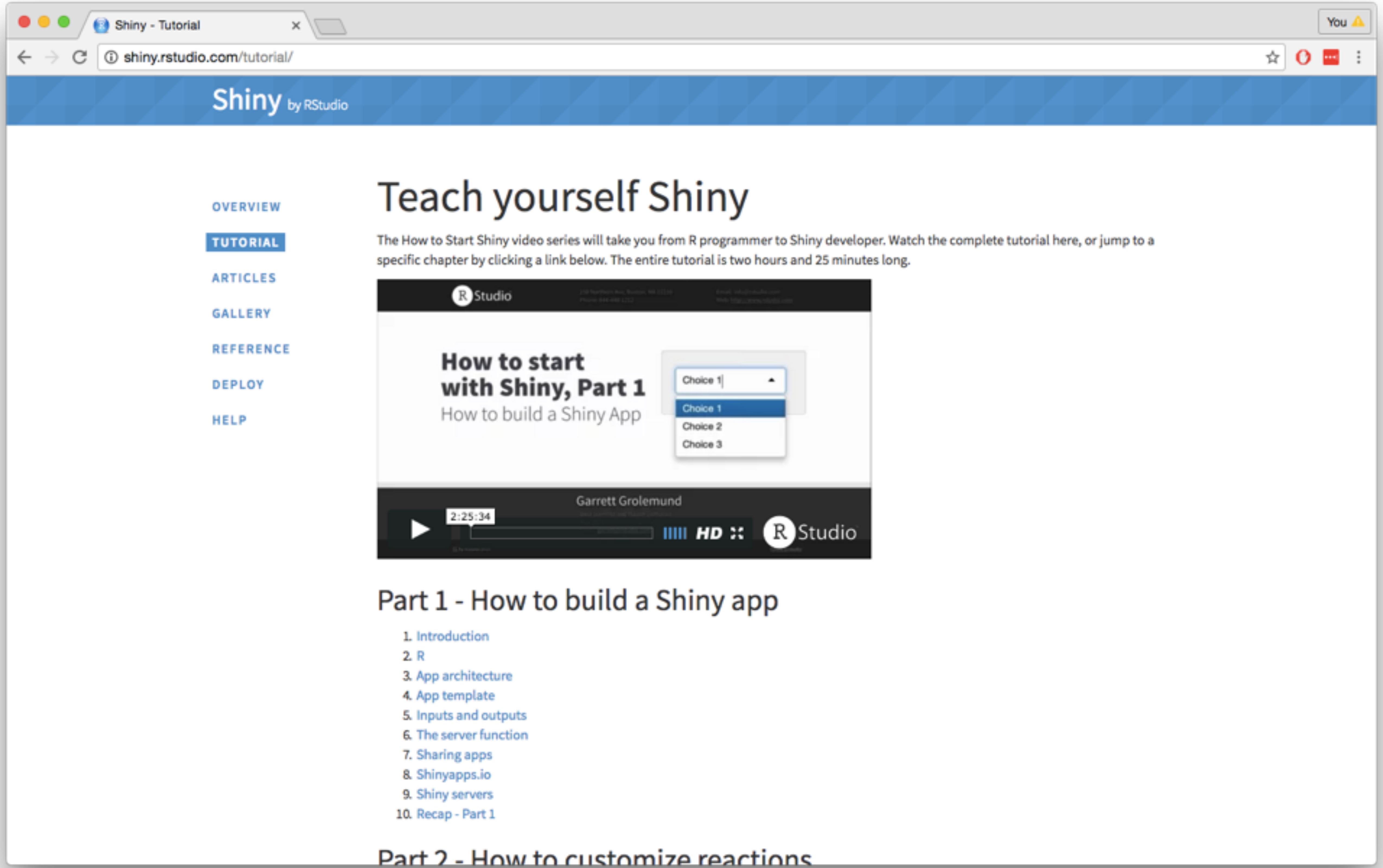
```
# Load shiny package
library(shiny)

# What web page looks like, make inputs
ui <- fluidPage(<your code>)

# How outputs are created/updated
server <- function(input, output){
            <your code>}

# Rendering of web app
shinyApp(ui = ui, server = server)
```

$$\begin{tikzcd} \text{ui} & \text{server} \\ \text{loads once} \arrow[u, bend left] & \text{loads many times} \arrow[u, bend right] \end{tikzcd}$$



Teach yourself Shiny

The How to Start Shiny video series will take you from R programmer to Shiny developer. Watch the complete tutorial here, or jump to a specific chapter by clicking a link below. The entire tutorial is two hours and 25 minutes long.

Part 1 - How to build a Shiny app

1. Introduction
2. R
3. App architecture
4. App template
5. Inputs and outputs
6. The server function
7. Sharing apps
8. Shinyapps.io
9. Shiny servers
10. Recap - Part 1

Part 2 - How to customize reactions

http://shiny.rstudio.com/tutorial/

http://shinyapps.io/

What Is Shiny Good for?	**What Is Shiny Bad for?**
small amounts of data	large amounts of data
single page websites	multi-page websites
speed not important	speed *is* important
fast prototyping	**full scale websites**

`ui` Code

```
selectInput(
                                                            )
  textInput
  tableInput
  sliderInput
```

```
selectInput(inputId = "year"
                                                            )
```

```
selectInput(inputId = "year",
            label ="Election Year"
                                                  )
```

```
selectInput(inputId = "year",
            label ="Election Year",
            choices =                         )
```

```
selectInput(inputId = "year",
            label ="Election Year",
           choices = c(levels(data_result$year))
```

```
plotOutput(                    )
plotOutput("result_map")
```

`server` Code

```
output
```

```
output$result_map
```

remember?

```
plotOutput("result_map")
```

```
output$result_map = renderPlot({











  })
```

```
output$result_map = renderPlot({
  ggplot(subset(data_plot, year == input$year),
         aes(x = long, y = lat, group = group,
             fill = party_winner)) +
    geom_polygon(color = "white") +
    scale_fill_manual(values = c("blue", “red",
               "yellow", "green")) +
    coord_map(projection = "polyconic") +
    theme_void() +
    theme(legend.position = “top",
           text = element_text(size = 40))
  })
```

any standard plot code

```
output$result_map = renderPlot({
  ggplot(subset(data_plot, year == input$year),
         aes(x = long, y = lat, group = group,
             fill = party_winner)) +
    geom_polygon(color = "white") +
    scale_fill_manual(values = c("blue", “red",
               "yellow", "green")) +
    coord_map(projection = "polyconic") +
    theme_void() +
    theme(legend.position = “top",
           text = element_text(size = 40))
  })
```

from

`selectInput(inputId = “year”,…)`

```
data_sum
```

```
data_sum = reactive({







  })
```

```
data_sum = reactive({
    data_result %>%
      filter(year == input$year) %>%
      group_by(party_winner) %>%
      summarise(total_votes =
            sum(num_electoral_votes,
                    na.rm = T)) %>%
      ungroup() %>%
      filter(!is.na(party_winner))
 })
```

any standard
R code

```
data_sum = reactive({
    data_result %>%
      filter(year == input$year) %>%
      group_by(party_winner) %>%
      summarise(total_votes =
            sum(num_electoral_votes,
                    na.rm = T)) %>%
      ungroup() %>%
      filter(!is.na(party_winner))
  })
```

from

```
selectInput(inputId = “year”,…)
```

Without a Reactive Value

```
output$result_table = renderTable({









  })
```

With a Reactive Value

```
output$result_table = renderTable({





  })
```

Without a Reactive Value

```
output$result_table = renderTable({
    data_result %>%
      filter(year == input$year) %>%
      group_by(party_winner) %>%
      summarise(total_votes =
          sum(num_electoral_votes, na.rm = T)) %>%
      ungroup() %>%
      filter(!is.na(party_winner))




  })
```

With a Reactive Value

```
output$result_table = renderTable({
    data_sum()




  })
```

Without a Reactive Value

```
output$result_table = renderTable({
    data_result %>%
      filter(year == input$year) %>%
      group_by(party_winner) %>%
      summarise(total_votes =
          sum(num_electoral_votes, na.rm = T)) %>%
      ungroup() %>%
      filter(!is.na(party_winner))




  })
```

With a Reactive Value

```
output$result_table = renderTable({
    data_sum()



  })
```

technically a function

Without a Reactive Value

```
output$result_table = renderTable({
    data_result %>%
      filter(year == input$year) %>%
      group_by(party_winner) %>%
      summarise(total_votes =
          sum(num_electoral_votes, na.rm = T)) %>%
      ungroup() %>%
      filter(!is.na(party_winner)) %>%
      rename(Party = party_winner) %>%
      rename("Electoral College Votes" =
               total_votes)
  })
```

With a Reactive Value

```
output$result_table = renderTable({
    data_sum() %>%
      rename(Party = party_winner) %>%
      rename("Electoral College Votes" =
               total_votes)
  })
```

`ui` HTML and CSS Code

```
ui <- fluidPage(









 selectInput(inputId = "year", label = "Election Year",
             choices = c(levels(factor(data_result$year)))),

 tableOutput("result_table"),

 textOutput("result_text"),

 plotOutput("result_map")

)
```

```
ui <- fluidPage(


  title = "R-Ladies Paris: Shiny Tutorial",
```

adds title to tab
on webpage

```
 selectInput(inputId = "year", label = "Election Year",
              choices = c(levels(factor(data_result$year))),

 tableOutput("result_table"),

 textOutput("result_text"),

 plotOutput("result_map")

)
```

```
ui <- fluidPage(


  title = "R-Ladies Paris: Shiny Tutorial",

  tags$h1("Historical United States Presidential Election
           Results"),
```

adds header to
top of webpage

```
 selectInput(inputId = "year", label = "Election Year",
              choices = c(levels(factor(data_result$year)))),

 tableOutput("result_table"),

 textOutput("result_text"),

 plotOutput("result_map")

)
```

```
ui <- fluidPage(



   title = "R-Ladies Paris: Shiny Tutorial",

   tags$h1("Historical United States Presidential Election
            Results"),
   tags$h4("Data from”,
           tags$a(href = “#",
                  "Wikipedia: List of United States presidential
                   election results by state”),
```

adds smaller header to top of webpage

```
  selectInput(inputId = "year", label = "Election Year",
                choices = c(levels(factor(data_result$year)))),

  tableOutput("result_table"),

  textOutput("result_text"),

  plotOutput("result_map")

)
```

```
ui <- fluidPage(



   title = "R-Ladies Paris: Shiny Tutorial",

   tags$h1("Historical United States Presidential Election
            Results"),
   tags$h4("Data from”,
           tags$a(href = “#",
                  "Wikipedia: List of United States presidential
                   election results by state”),
```

adds a link

```
  selectInput(inputId = "year", label = "Election Year",
               choices = c(levels(factor(data_result$year)))),

  tableOutput("result_table"),

  textOutput("result_text"),

  plotOutput("result_map")

)
```

```
ui <- fluidPage(



  title = "R-Ladies Paris: Shiny Tutorial",

  tags$h1("Historical United States Presidential Election
           Results"),
  tags$h4("Data from”,
          tags$a(href = “#",
                 "Wikipedia: List of United States presidential
                  election results by state”),
```

use to call HTML tags

```
 selectInput(inputId = "year", label = "Election Year",
             choices = c(levels(factor(data_result$year)))),

 tableOutput("result_table"),

 textOutput("result_text"),

 plotOutput("result_map")

)
```

```
ui <- fluidPage(

  theme = "bootswatch-cerulean.css",

  title = "R-Ladies Paris: Shiny Tutorial",

  tags$h1("Historical United States Presidential Election
           Results"),
  tags$h4("Data from”,
          tags$a(href = “#",
                "Wikipedia: List of United States presidential
                 election results by state”),
```

adds CSS template in “www” folder

```
 selectInput(inputId = "year", label = "Election Year",
             choices = c(levels(factor(data_result$year)))),

 tableOutput("result_table"),

 textOutput("result_text"),

 plotOutput("result_map")

)
```