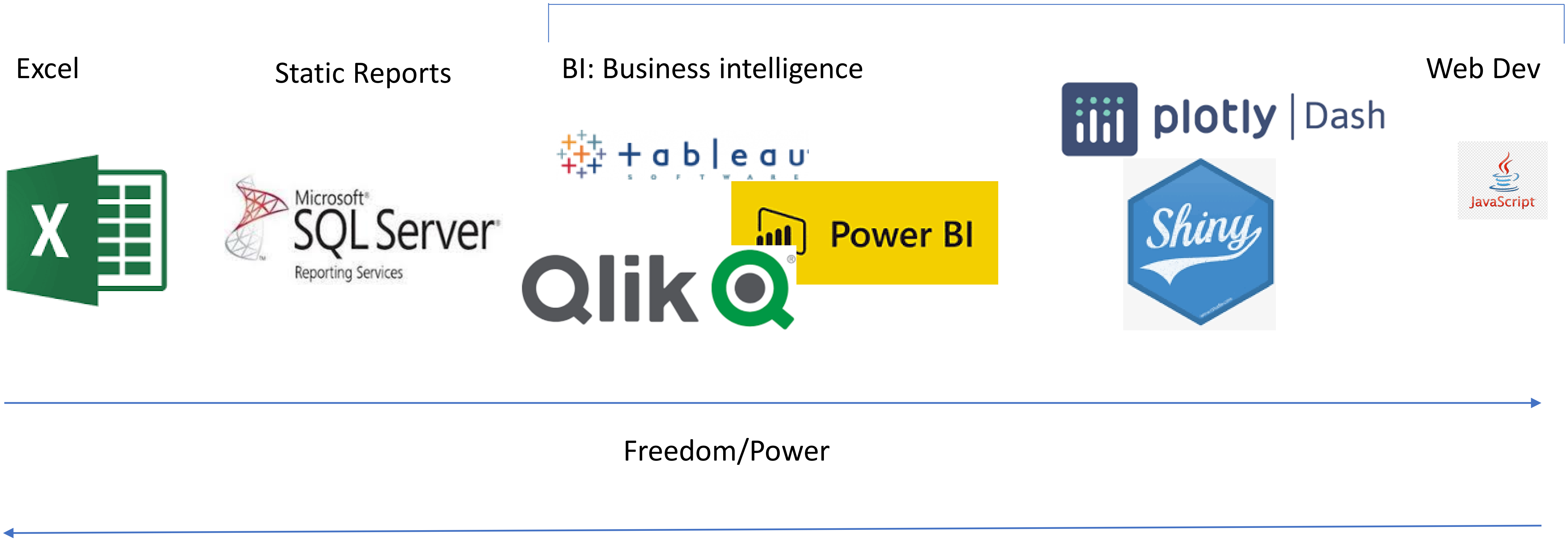
R Shiny



«Interactive Reports»

Excel

Static Reports

BI: Business intelligence

Web Dev

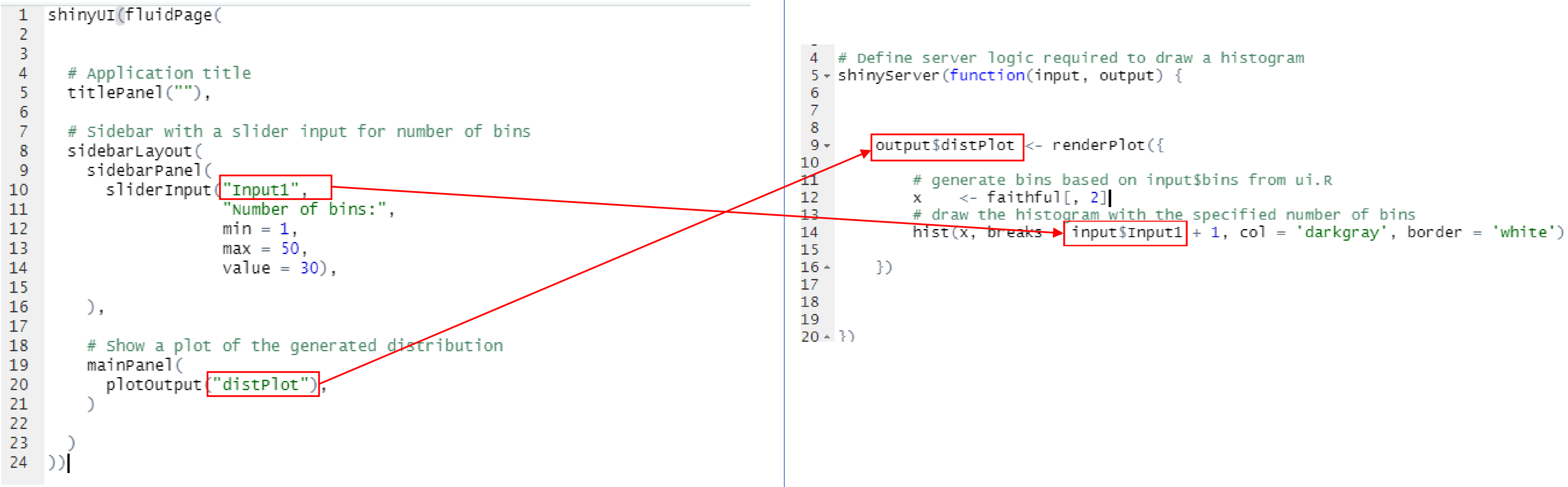
Freedom/Power

Number of Users & Simplicity

Your best friends

Package Dokumentation

RShiny Official Page



Ui.R

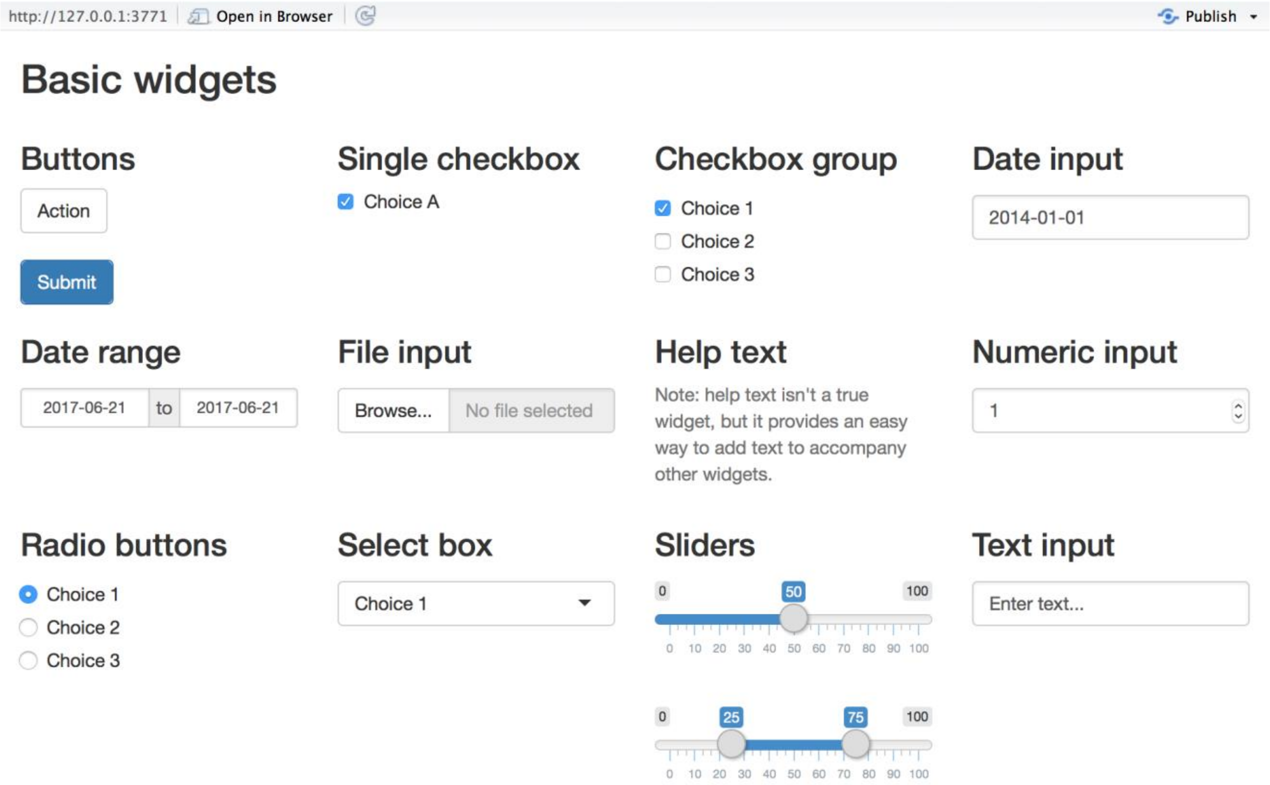
```
shinyUI(fluidPage(


  # Application title
  titlePanel(""),

  # Sidebar with a slider input for number of bins
  sidebarLayout(
    sidebarPanel(
      sliderInput("Input1",
                  "Number of bins:",
                  min = 1,
                  max = 50,
                  value = 30),

    ),

    # Show a plot of the generated distribution
    mainPanel(
      plotOutput("distPlot"),
    )

  )
))
```

Server.R

```
# Define server logic required to draw a histogram
shinyServer(function(input, output) {



  output$distPlot <- renderPlot({

    # generate bins based on input$bins from ui.R
    x    <- faithful[, 2]
    # draw the histogram with the specified number of bins
    hist(x, breaks = input$Input1 + 1, col = 'darkgray', border = 'white')

  })




})
```

Input Options - General

input-action.R
input-checkbox.R
input-checkboxgroup.R
input-date.R
input-daterange.R
input-file.R
input-numeric.R
input-password.R
input-radiobuttons.R
input-select.R
input-slider.R
input-submit.R
input-text.R
input-textarea.R
input-utils.R



Basic widgets

Buttons

Single checkbox

☑ Choice A

Checkbox group

☑ Choice 1
☐ Choice 2
☐ Choice 3

Date input

2014-01-01

Date range

2017-06-21 to 2017-06-21

File input

Help text

Note: help text isn't a true widget, but it provides an easy way to add text to accompany other widgets.

Numeric input

1

Radio buttons

● Choice 1
○ Choice 2
○ Choice 3

Select box

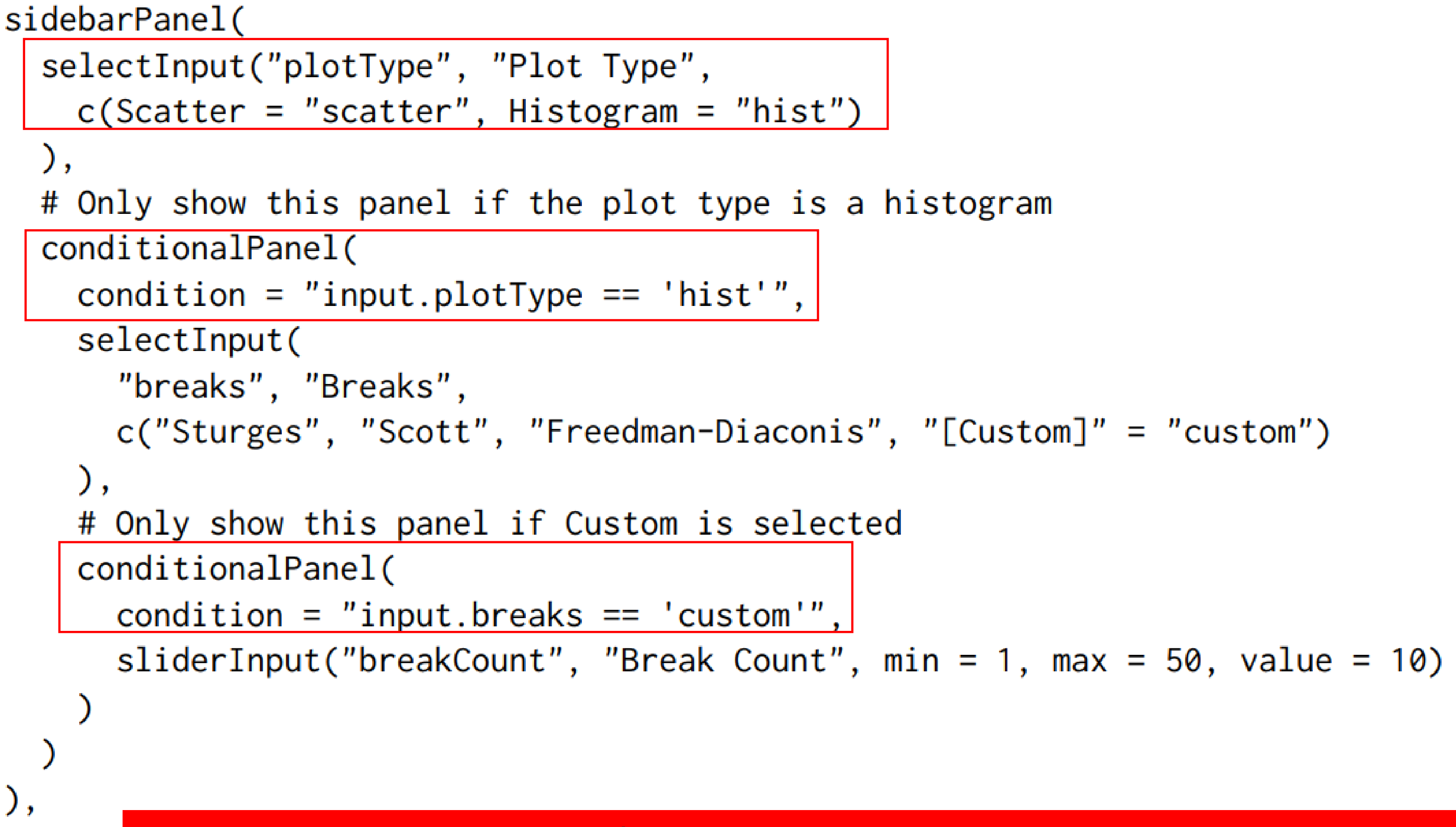
Choice 1

Sliders

Text input

Enter text...

Input Options - conditionalPanel

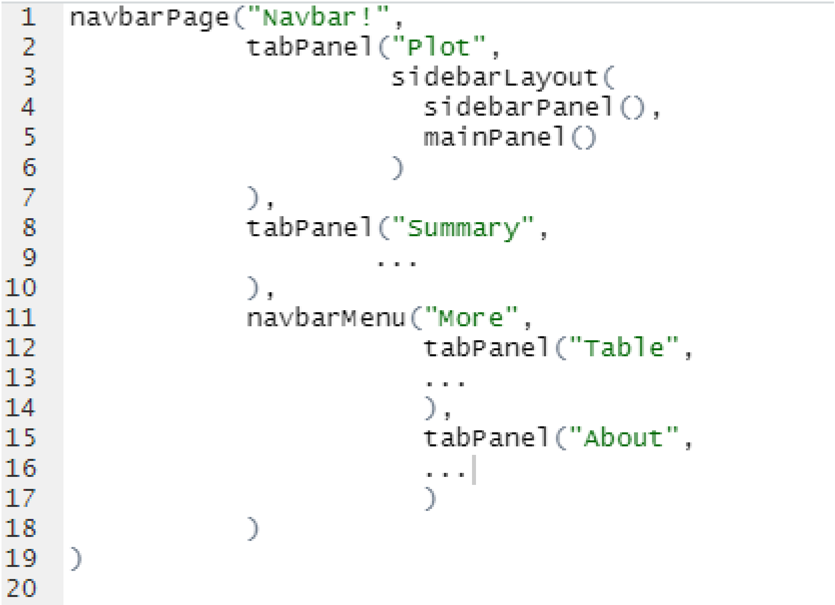


```
sidebarPanel(
  selectInput("plotType", "Plot Type",
    c(Scatter = "scatter", Histogram = "hist")
  ),
  # Only show this panel if the plot type is a histogram
  conditionalPanel(
    condition = "input.plotType == 'hist'",
    selectInput(
      "breaks", "Breaks",
      c("Sturges", "Scott", "Freedman-Diaconis", "[Custom]" = "custom")
    ),
    # Only show this panel if Custom is selected
    conditionalPanel(
      condition = "input.breaks == 'custom'",
      sliderInput("breakCount", "Break Count", min = 1, max = 50, value = 10)
    )
  )
),
```

Condition: A JavaScript condition!

Input Options - NavBar

Exchange fluidPage by navbarPage



```
navbarPage("Navbar!",
           tabPanel("Plot",
                    sidebarLayout(
                      sidebarPanel(),
                      mainPanel()
                    )
           ),
           tabPanel("Summary",
                   ...
           ),
           navbarMenu("More",
                      tabPanel("Table",
                      ...
                      ),
                      tabPanel("About",
                      ...
                      )
           )
)

```

Link

Reactiviy (in server)

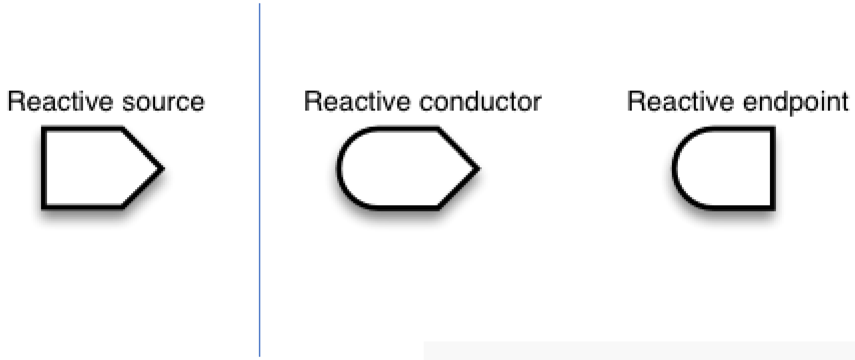
```
output$hist=hist(c(1,2,3,4),breaks=input$n)


output$hist =renderplot({hist(c(1,2,3,4),breaks=input$n)})
```

```
a=function(input$variable)


a=reactive({function(input$varaible]})
```

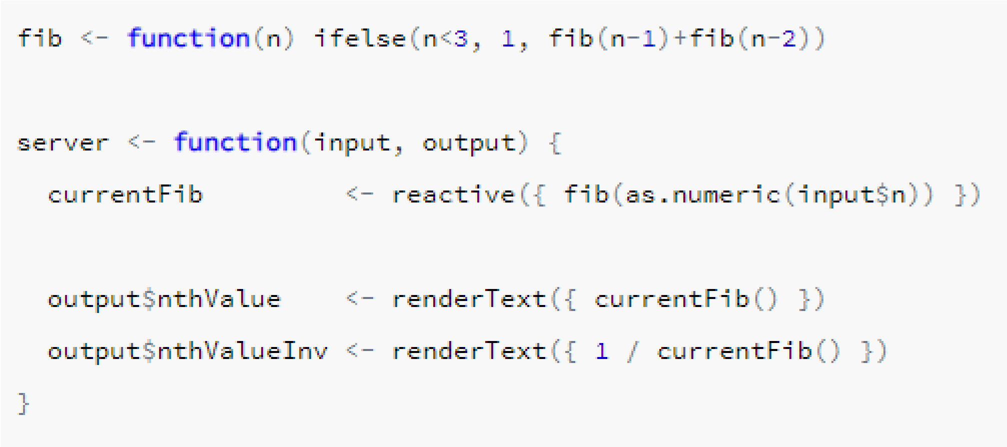
renderPlot
renderPrint
renderTable
renderText

Reactiviy

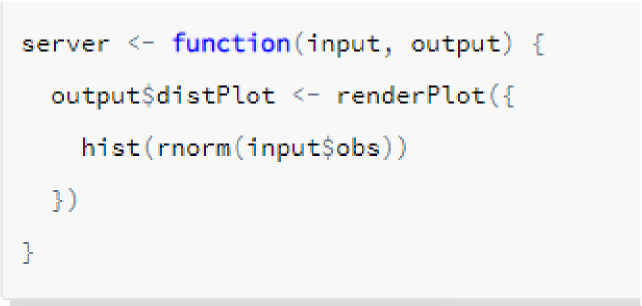
Ui



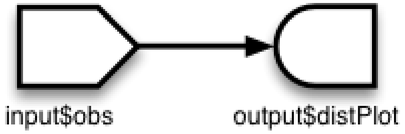
Server

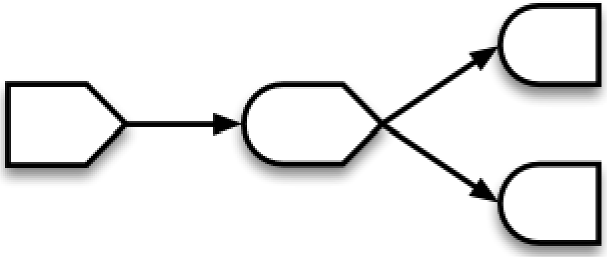


```
fib <- function(n) ifelse(n<3, 1, fib(n-1)+fib(n-2))

server <- function(input, output) {
  currentFib         <- reactive({ fib(as.numeric(input$n)) })

  output$nthValue    <- renderText({ currentFib() })
  output$nthValueInv <- renderText({ 1 / currentFib() })
}
```



```
server <- function(input, output) {
  output$distPlot <- renderPlot({
    hist(rnorm(input$obs))
  })
}
```





Execution scheduling of reactive Elements is random

Comparison

RShiny VS BI

+ Powerfull Backend with all functionality of R
- Harder to productionise
- Performance needs more work
- Filters not dynamic

- Only can Aggregate by Dimensions
+ Easy to deploy, monitor, manage etc
+ Scalable without effort