Introduction to Shiny

Sam Bashevkin

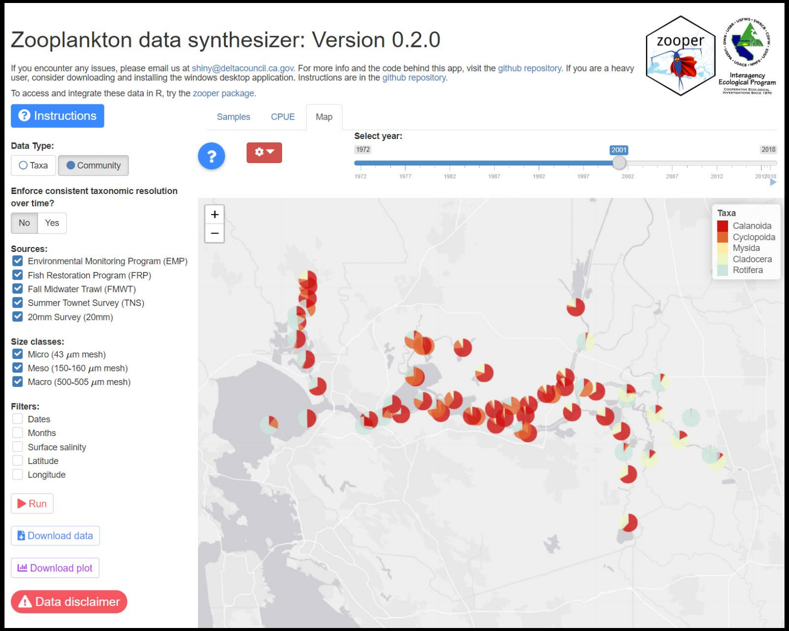
State Water Resources Control Board

What is shiny?

- R package to build interactive apps with a graphical user interface.
- Practically any R functionality (including external packages) can be used in a shiny app.
- Apps can be hosted online and distributed as web pages, or used locally

Why Shiny?

- Bring R functionality to non-R-users.
- Communicate science.
- Toggle parameters and quickly visualize the effects.



Zooplankton data synthesizer: Version 0.2.0

If you encounter any issues, please email us at shiny@deltacouncil.ca.gov. For more info and the code behind this app, visit the github repository. If you are a heavy user, consider downloading and installing the windows desktop application. Instructions are in the github repository.

To access and integrate these data in R, try the zooper package.

Data Type: Taxa [✓ Community]

Enforce consistent taxonomic resolution over time? [✓ No] Yes

Sources:
- ✓ Environmental Monitoring Program (EMP)
- ✓ Fish Restoration Program (FRP)
- ✓ Fall Midwater Trawl (FMWT)
- ✓ Summer Townet Survey (TNS)
- ✓ 20mm Survey (20mm)

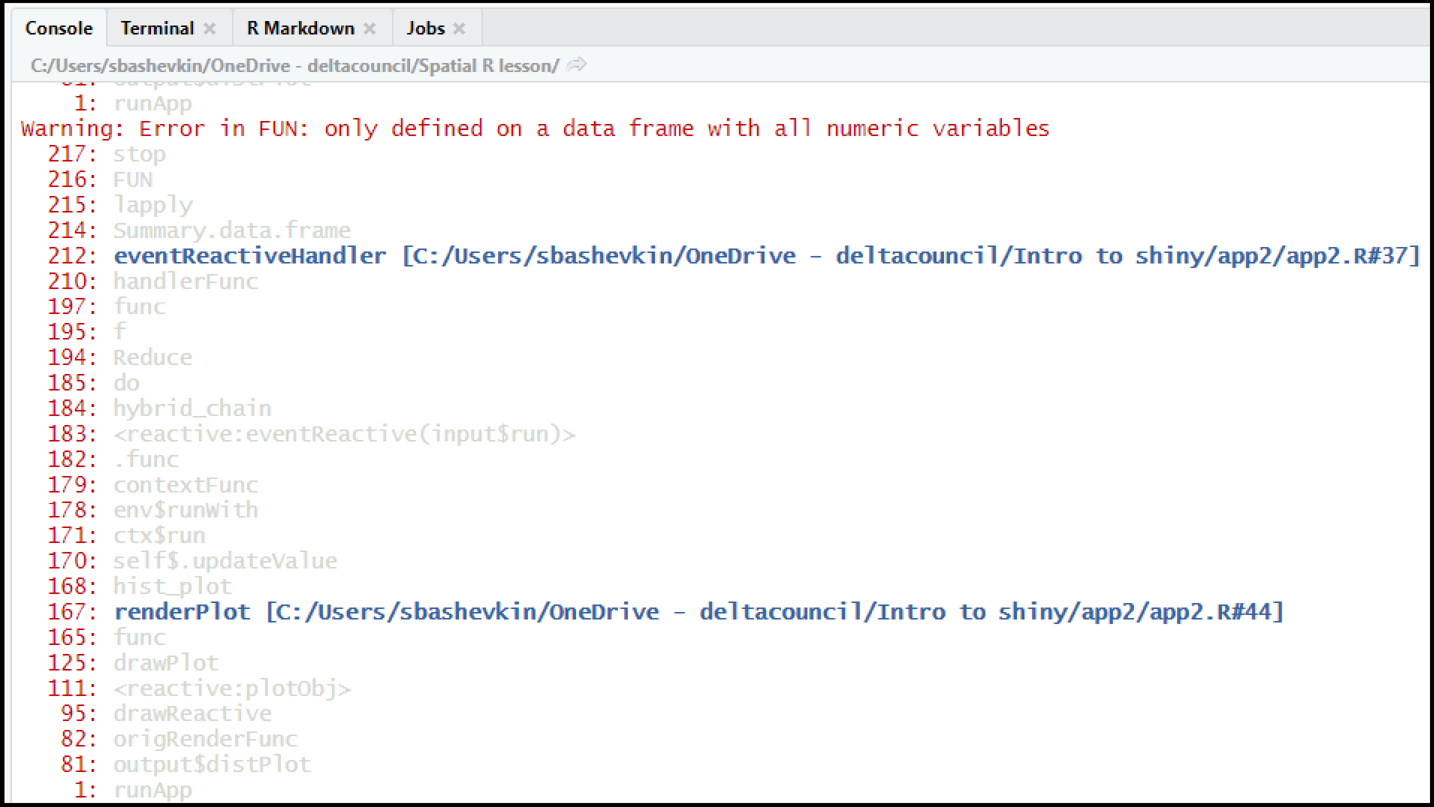
Size classes:
- ✓ Micro (43 μm mesh)
- ✓ Meso (150-160 μm mesh)
- ✓ Macro (500-505 μm mesh)

Filters:
- ☐ Dates
- ☐ Months
- ☐ Surface salinity
- ☐ Latitude
- ☐ Longitude

Select year: 2001

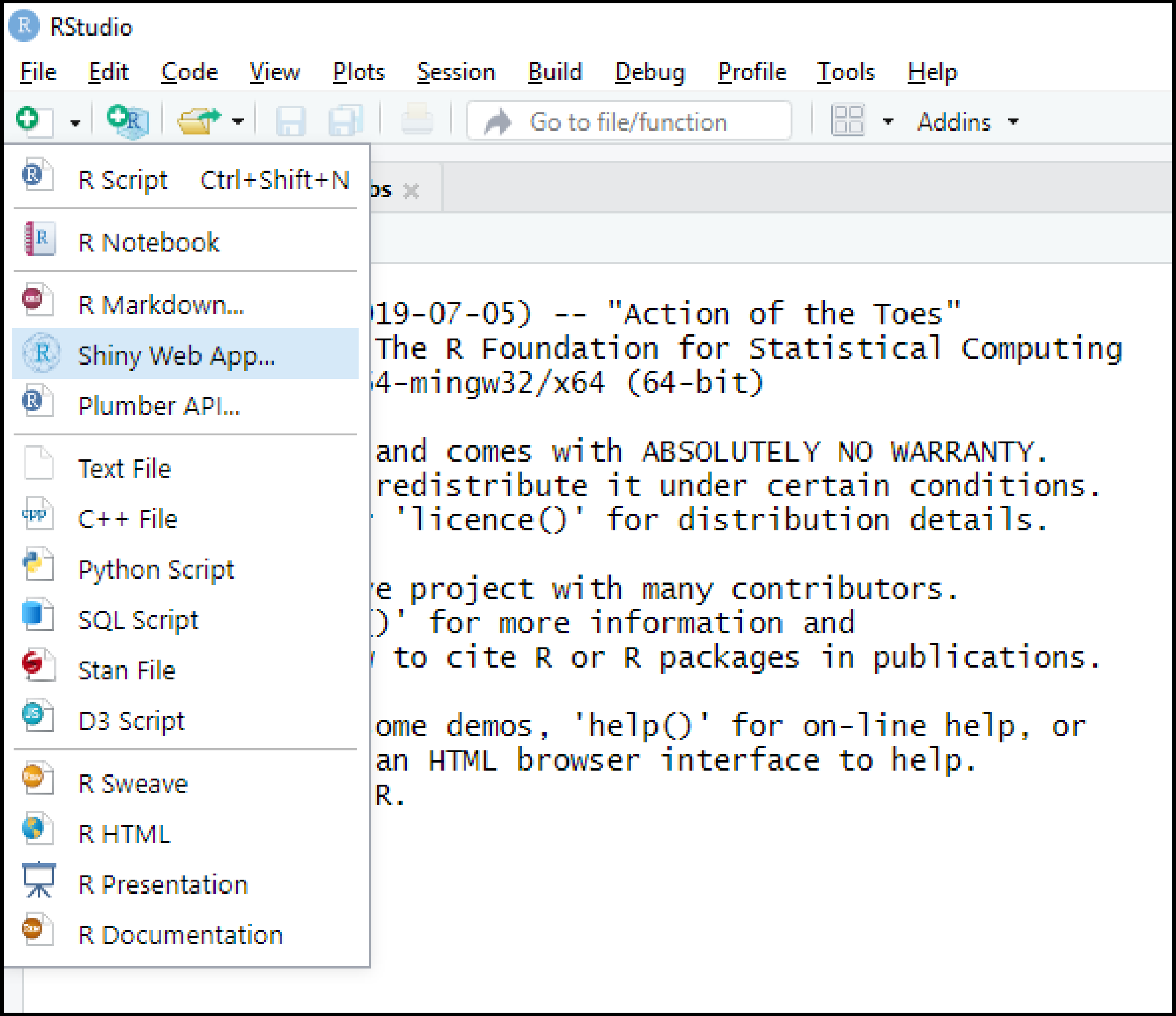
Why not shiny?

- Reduce your frustration load
- Show off your impressive coding skills
- Build something you will not have the time to maintain


Console
Terminal
R Markdown
Jobs
C:/Users/sbashevkin/OneDrive - deltacouncil/Spatial R lesson/

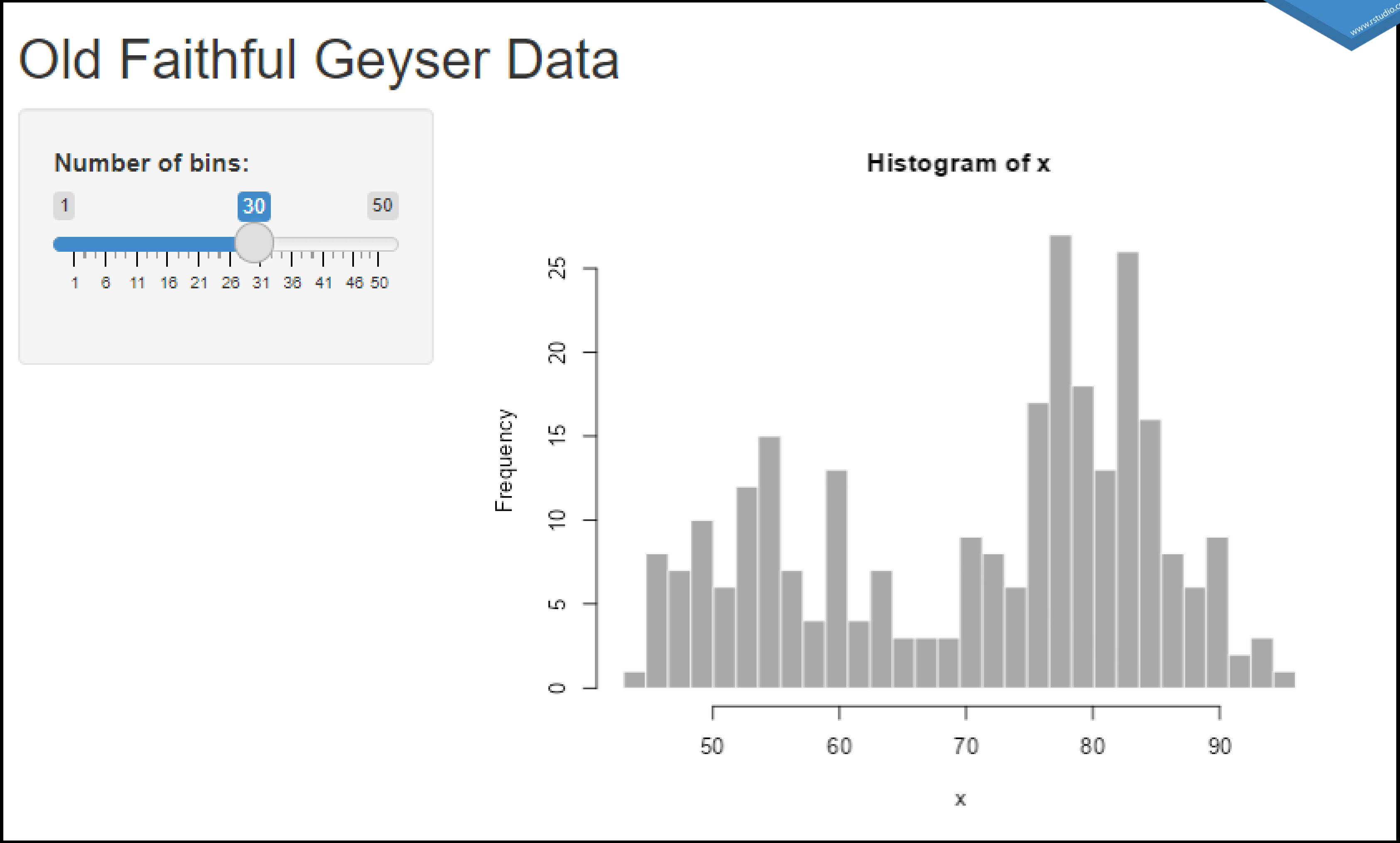
```
     1: runApp
Warning: Error in FUN: only defined on a data frame with all numeric variables
   217: stop
   216: FUN
   215: lapply
   214: Summary.data.frame
   212: eventReactiveHandler [C:/Users/sbashevkin/OneDrive - deltacouncil/Intro to shiny/app2/app2.R#37]
   210: handlerFunc
   197: func
   195: f
   194: Reduce
   185: do
   184: hybrid_chain
   183: <reactive:eventReactive(input$run)>
   182: .func
   179: contextFunc
   178: env$runWith
   171: ctx$run
   170: self$.updateValue
   168: hist_plot
   167: renderPlot [C:/Users/sbashevkin/OneDrive - deltacouncil/Intro to shiny/app2/app2.R#44]
   165: func
   125: drawPlot
   111: <reactive:plotObj>
    95: drawReactive
    82: origRenderFunc
    81: output$distPlot
     1: runApp
```

Basic shiny app

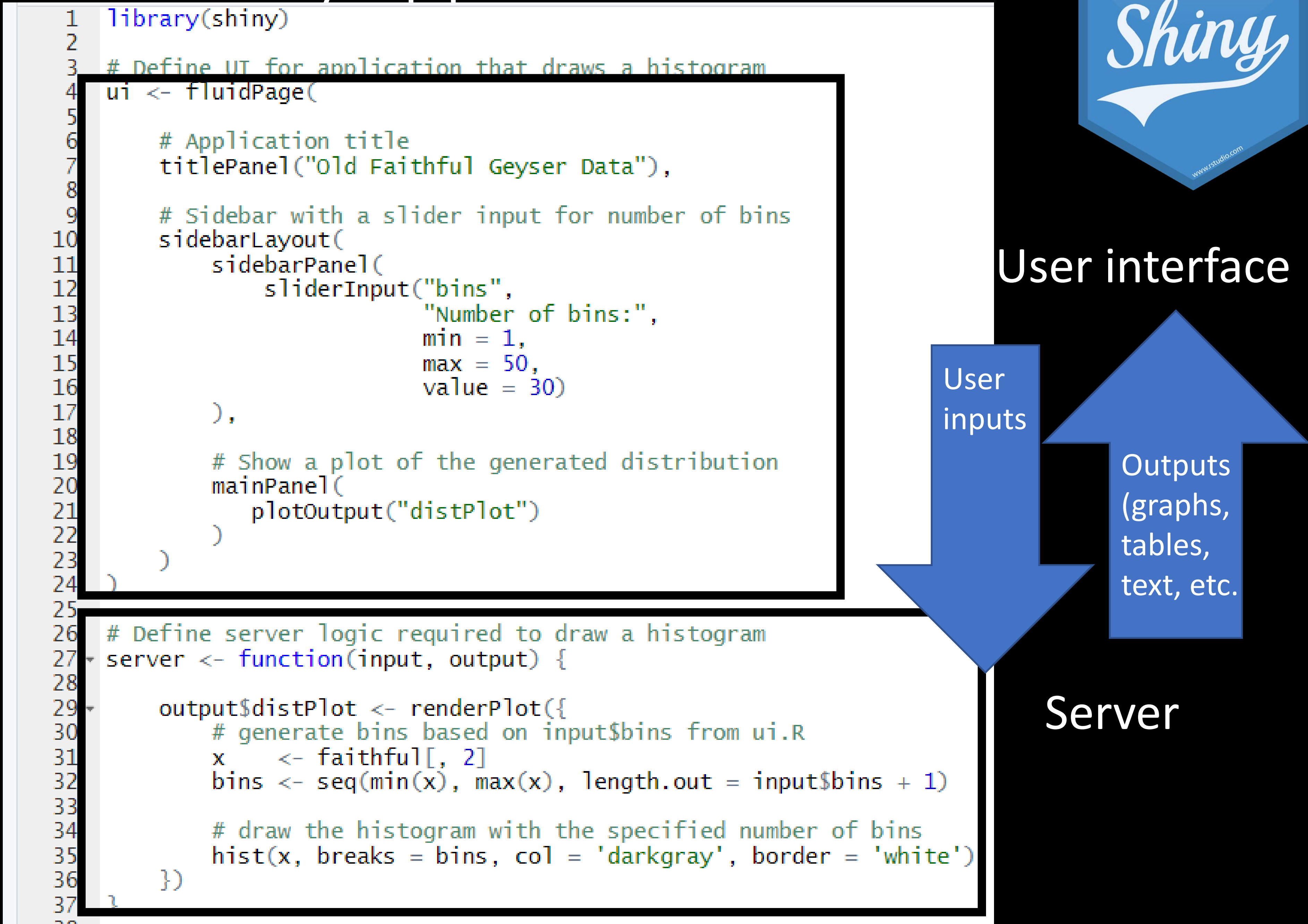


```
19-07-05) -- "Action of the Toes"
The R Foundation for Statistical Computing
4-mingw32/x64 (64-bit)

and comes with ABSOLUTELY NO WARRANTY.
redistribute it under certain conditions.
 'licence()' for distribution details.

e project with many contributors.
)' for more information and
 to cite R or R packages in publications.

ome demos, 'help()' for on-line help, or
an HTML browser interface to help.
R.
```

Basic shiny app

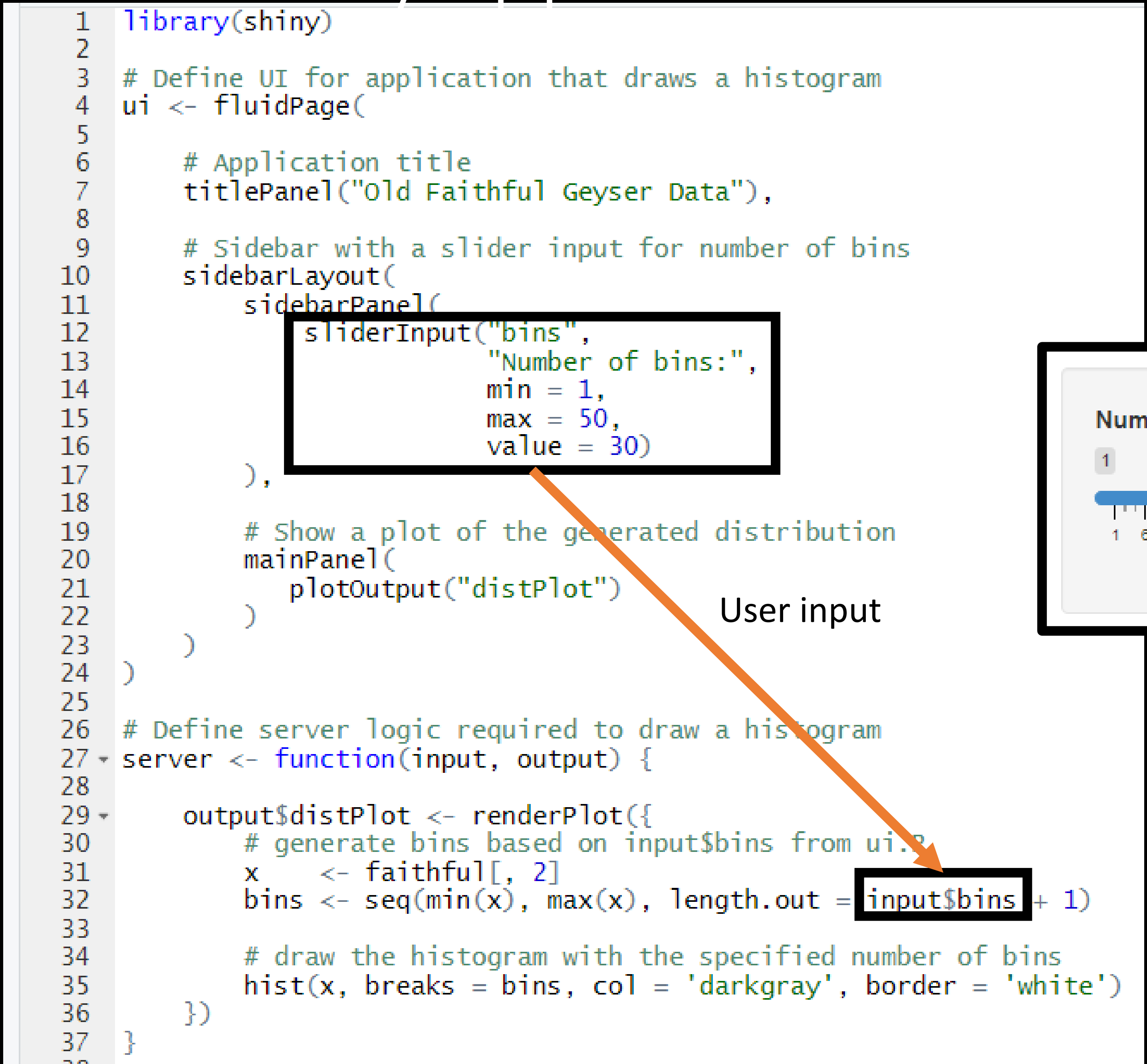


Basic shiny app



```r
library(shiny)

# Define UI for application that draws a histogram
ui <- fluidPage(

    # Application title
    titlePanel("Old Faithful Geyser Data"),

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

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

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

    output$distPlot <- renderPlot({
        # generate bins based on input$bins from ui.R
        x    <- faithful[, 2]
        bins <- seq(min(x), max(x), length.out = input$bins + 1)

        # draw the histogram with the specified number of bins
        hist(x, breaks = bins, col = 'darkgray', border = 'white')
    })
}
```

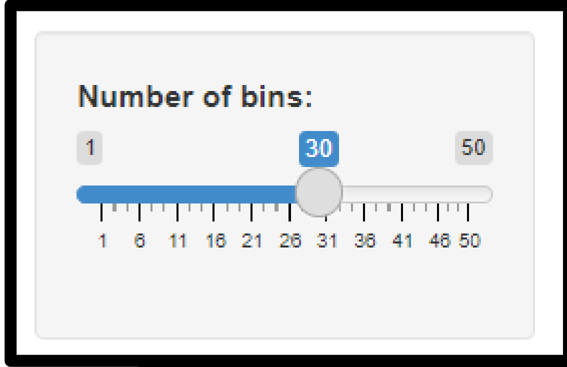
Basic shiny app



```r
library(shiny)

# Define UI for application that draws a histogram
ui <- fluidPage(

    # Application title
    titlePanel("Old Faithful Geyser Data"),

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

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

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

    output$distPlot <- renderPlot({
        # generate bins based on input$bins from ui.R
        x    <- faithful[, 2]
        bins <- seq(min(x), max(x), length.out = input$bins + 1)

        # draw the histogram with the specified number of bins
        hist(x, breaks = bins, col = 'darkgray', border = 'white')
    })
}
```

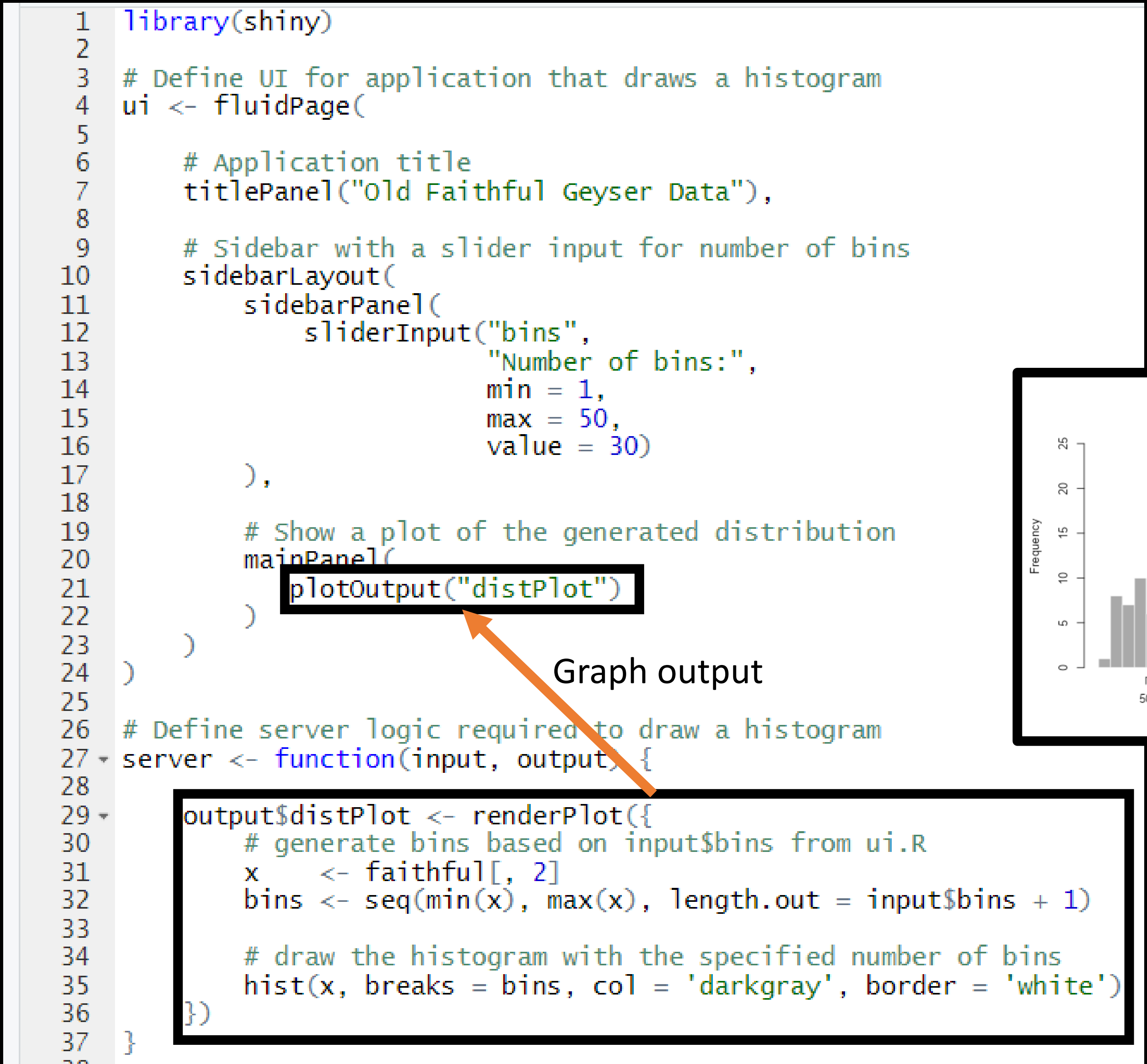
User input

User interface



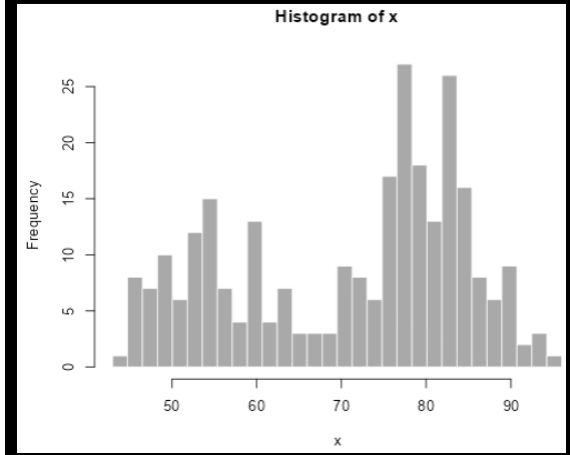
Server

Basic shiny app



```r
library(shiny)

# Define UI for application that draws a histogram
ui <- fluidPage(

    # Application title
    titlePanel("Old Faithful Geyser Data"),

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

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

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

   output$distPlot <- renderPlot({
        # generate bins based on input$bins from ui.R
        x    <- faithful[, 2]
        bins <- seq(min(x), max(x), length.out = input$bins + 1)

        # draw the histogram with the specified number of bins
        hist(x, breaks = bins, col = 'darkgray', border = 'white')
   })
}
```

User interface

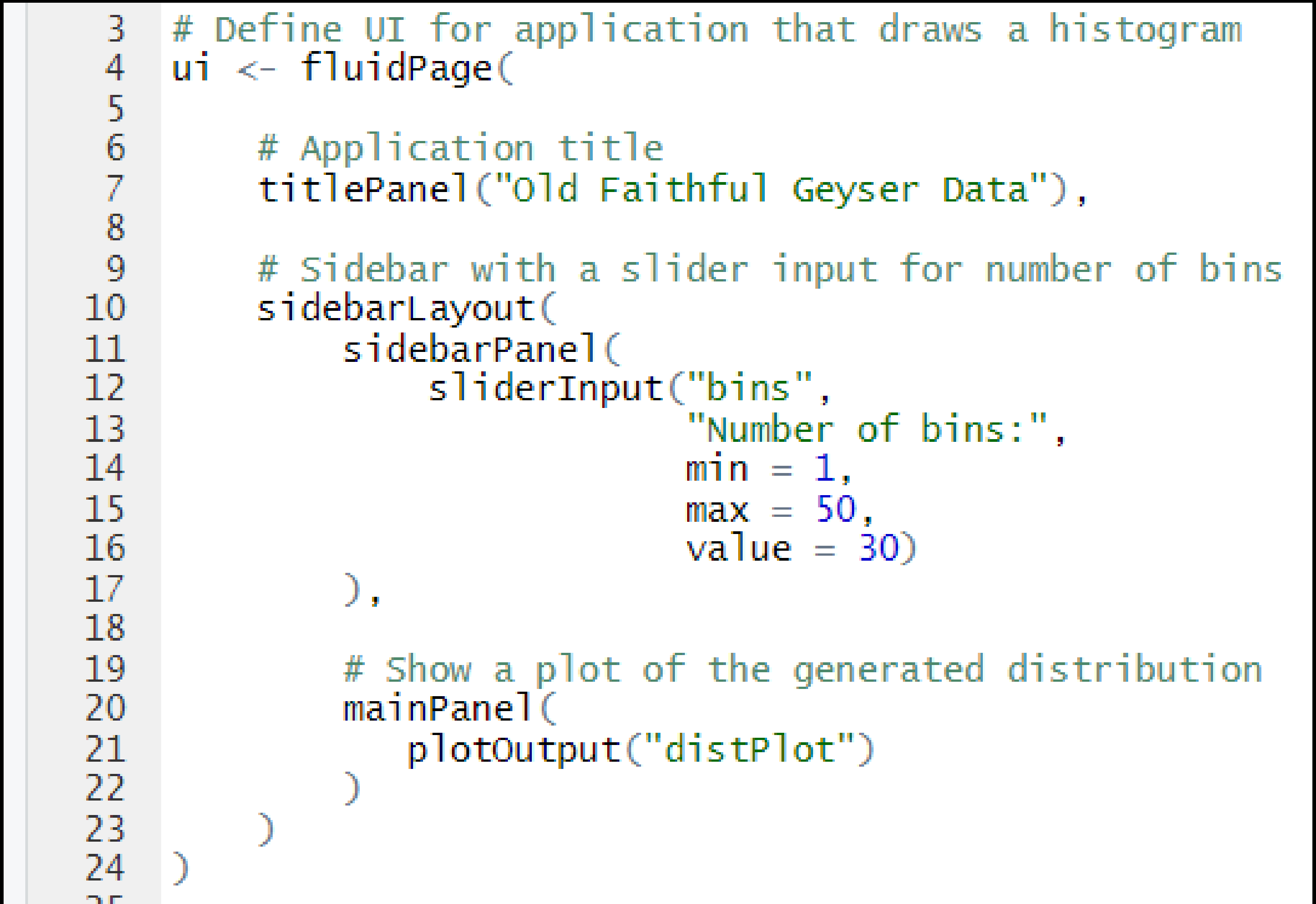


Graph output

Server

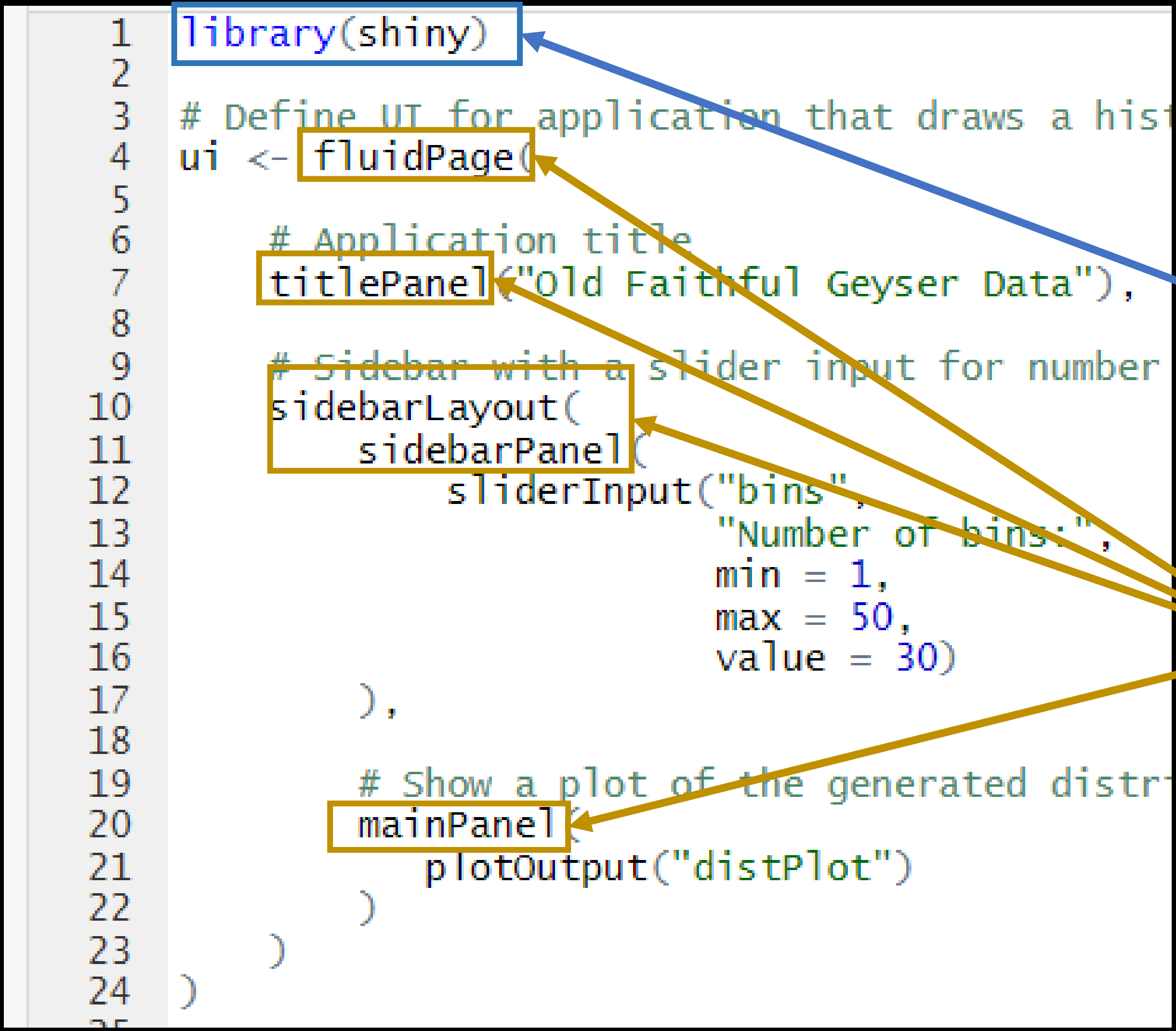
Shiny mechanics

The UI portion is analogous to a list of user-viewable components, wrapped in functions to set the layout



```
# Define UI for application that draws a histogram
ui <- fluidPage(

    # Application title
    titlePanel("Old Faithful Geyser Data"),

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

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

Shiny mechanics



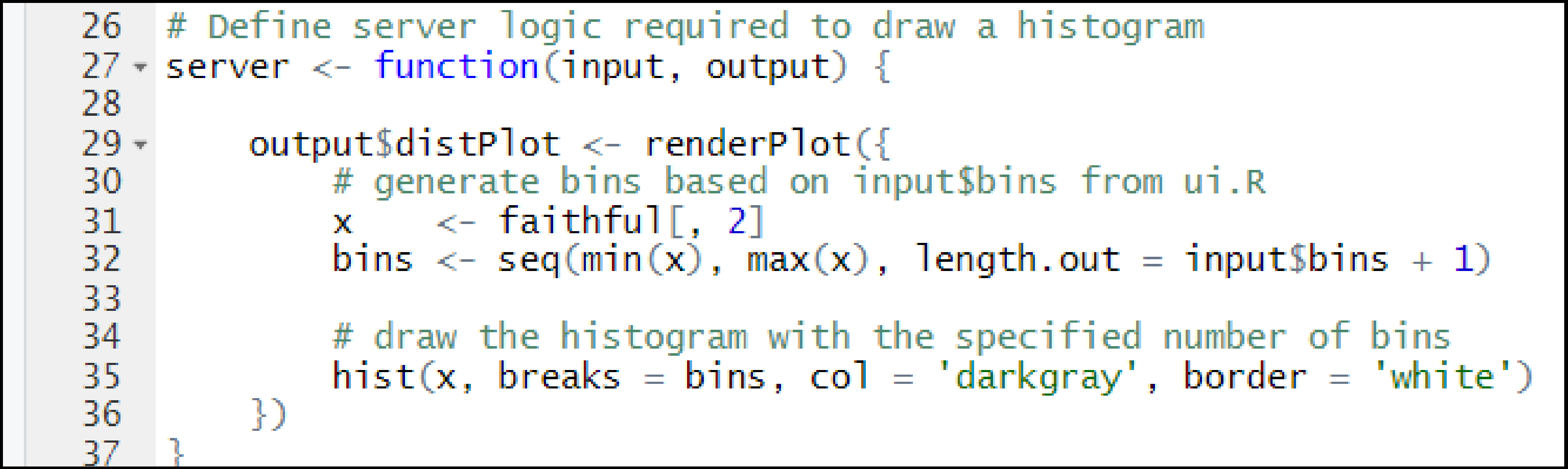
```
library(shiny)

# Define UI for application that draws a hist
ui <- fluidPage(

    # Application title
    titlePanel("Old Faithful Geyser Data"),

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

        # Show a plot of the generated distr
        mainPanel(
           plotOutput("distPlot")
        )
    )
)
```

Load things used by entire app (packages, functions, etc.)

Control layout

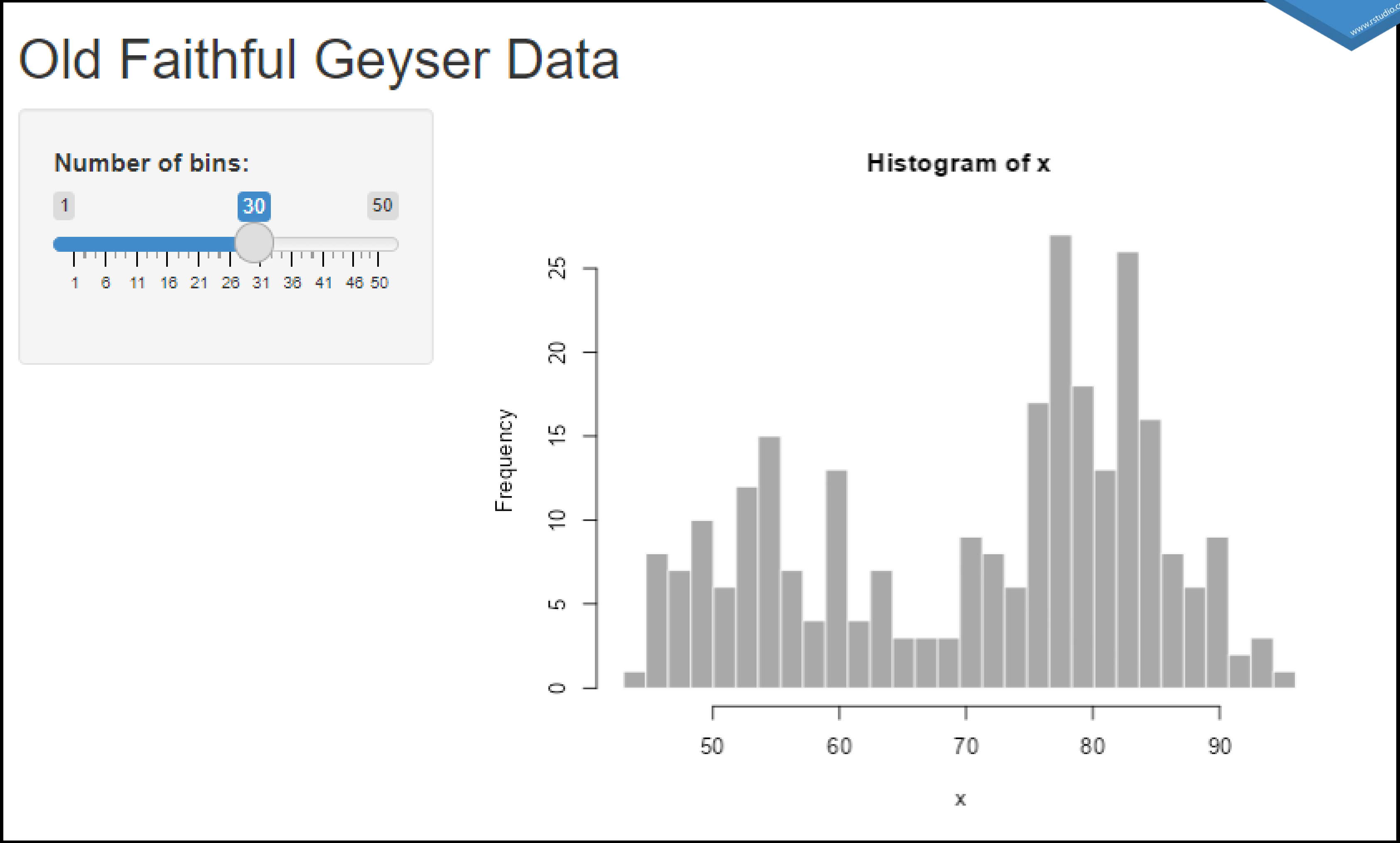
Shiny mechanics

The server portion is analogous to a series of functions with the inputs from the UI as arguments

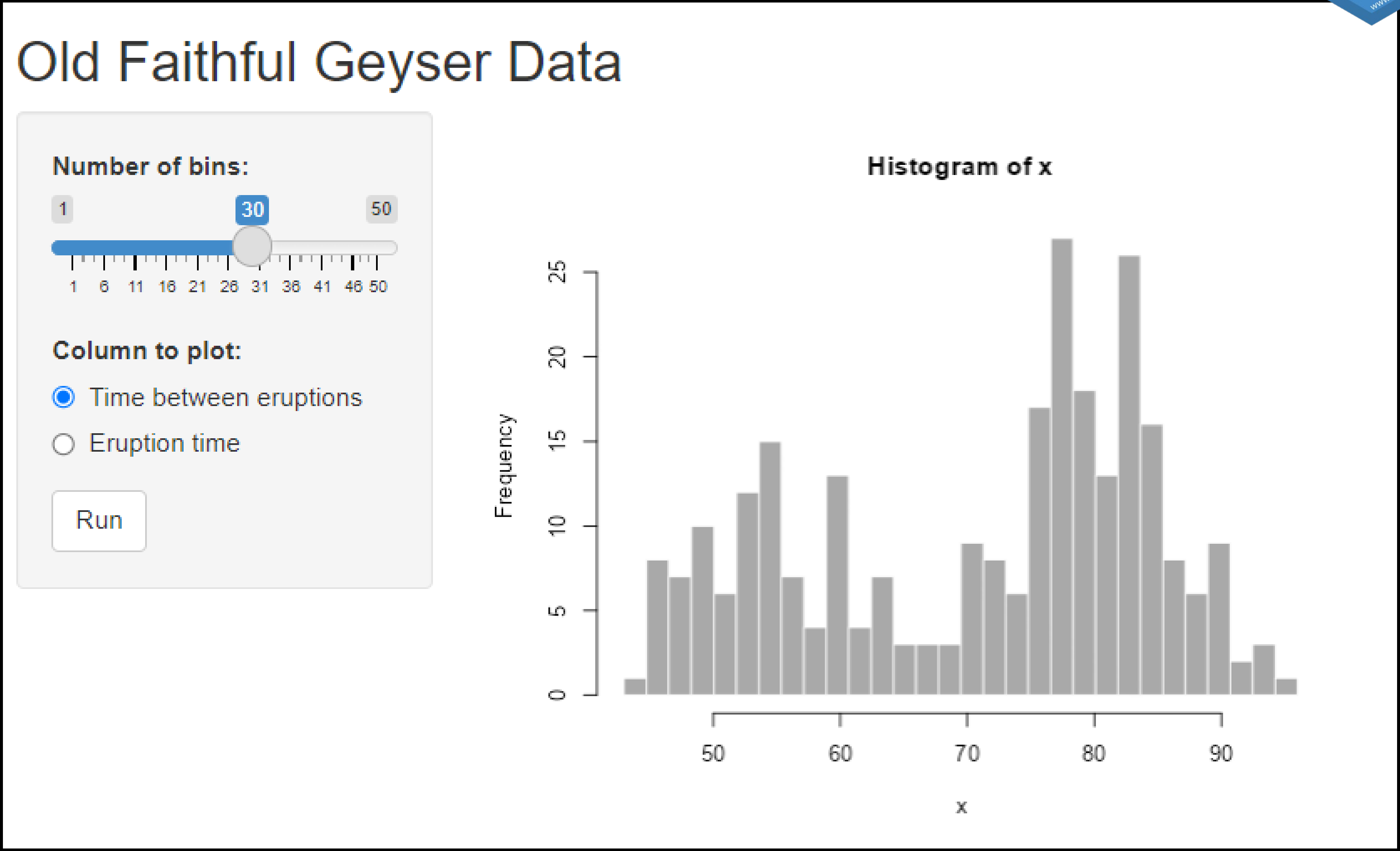


```r
# Define server logic required to draw a histogram
server <- function(input, output) {

    output$distPlot <- renderPlot({
        # generate bins based on input$bins from ui.R
        x    <- faithful[, 2]
        bins <- seq(min(x), max(x), length.out = input$bins + 1)

        # draw the histogram with the specified number of bins
        hist(x, breaks = bins, col = 'darkgray', border = 'white')
    })
}
```

Basic shiny app



More complex functionality



Old Faithful Geyser Data

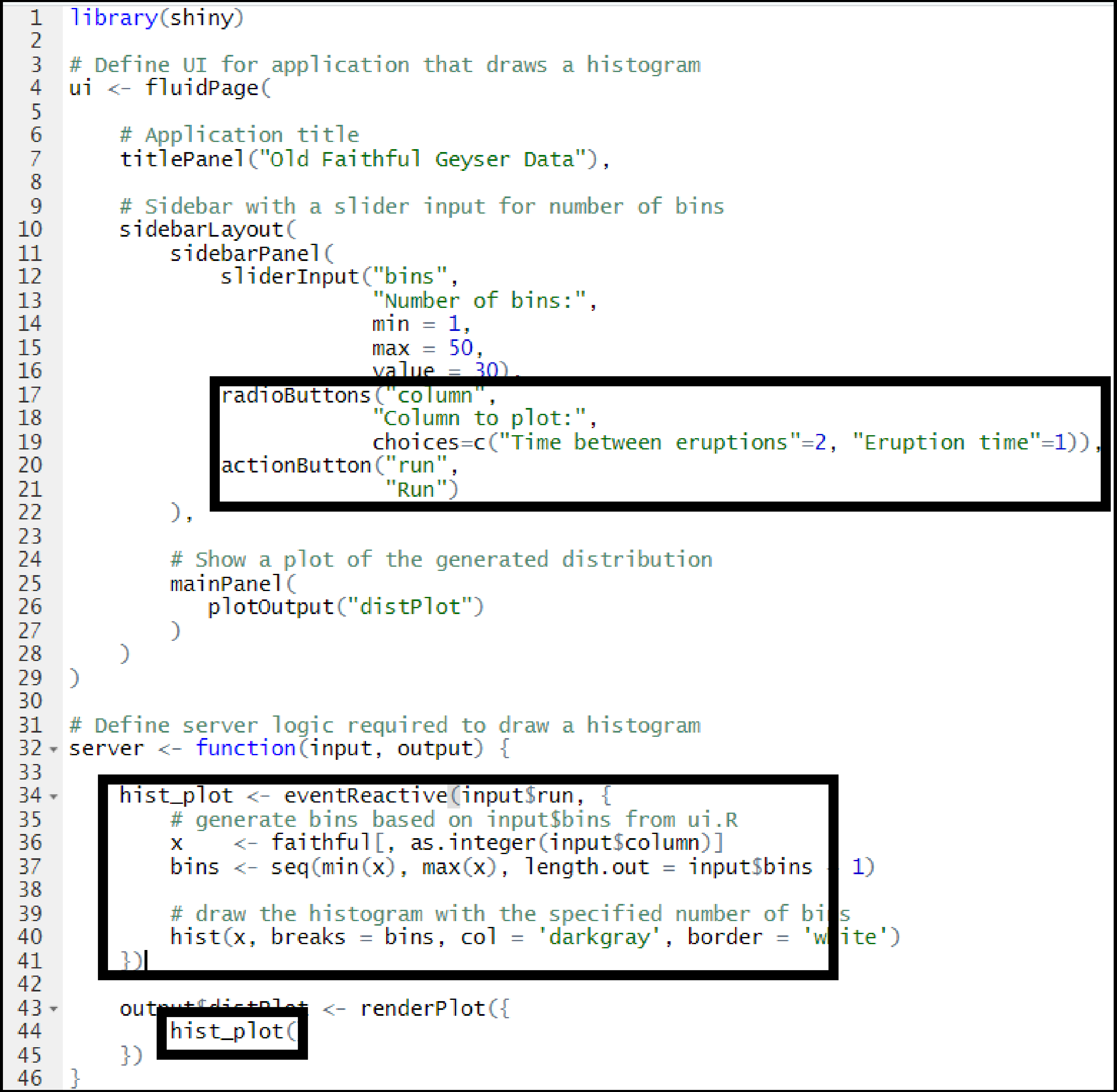
Number of bins: 30 (range 1–50)

Column to plot:
- Time between eruptions [✓]
- Eruption time

More complex functionality

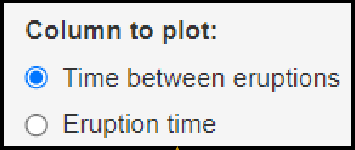
- Use *reactive* to create intermediate values that will be triggered by any change in their dependencies (inputs).
  - If you don't want everything to update as soon as the input changes, you can use *eventReactive*
  - Useful when the user must choose multiple inputs and you don't want a process to trigger until they have made all their selections.

More complex functionality



```r
library(shiny)

# Define UI for application that draws a histogram
ui <- fluidPage(

    # Application title
    titlePanel("Old Faithful Geyser Data"),

    # Sidebar with a slider input for number of bins
    sidebarLayout(
        sidebarPanel(
            sliderInput("bins",
                        "Number of bins:",
                        min = 1,
                        max = 50,
                        value = 30),
            radioButtons("column",
                        "Column to plot:",
                        choices=c("Time between eruptions"=2, "Eruption time"=1)),
            actionButton("run",
                         "Run")
        ),

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

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

    hist_plot <- eventReactive(input$run, {
        # generate bins based on input$bins from ui.R
        x    <- faithful[, as.integer(input$column)]
        bins <- seq(min(x), max(x), length.out = input$bins + 1)

        # draw the histogram with the specified number of bi s
        hist(x, breaks = bins, col = 'darkgray', border = 'w ite')
    })

    output$distPlot <- renderPlot({
        hist_plot(
    })
}
```

More complex functionality

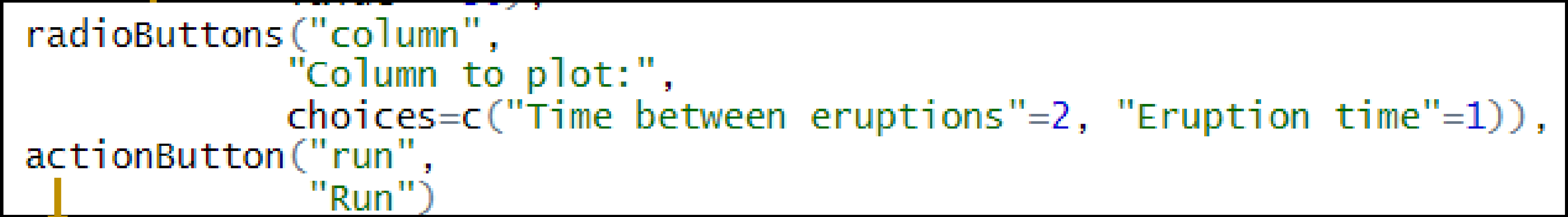


Column to plot:

[✓ Time between eruptions]

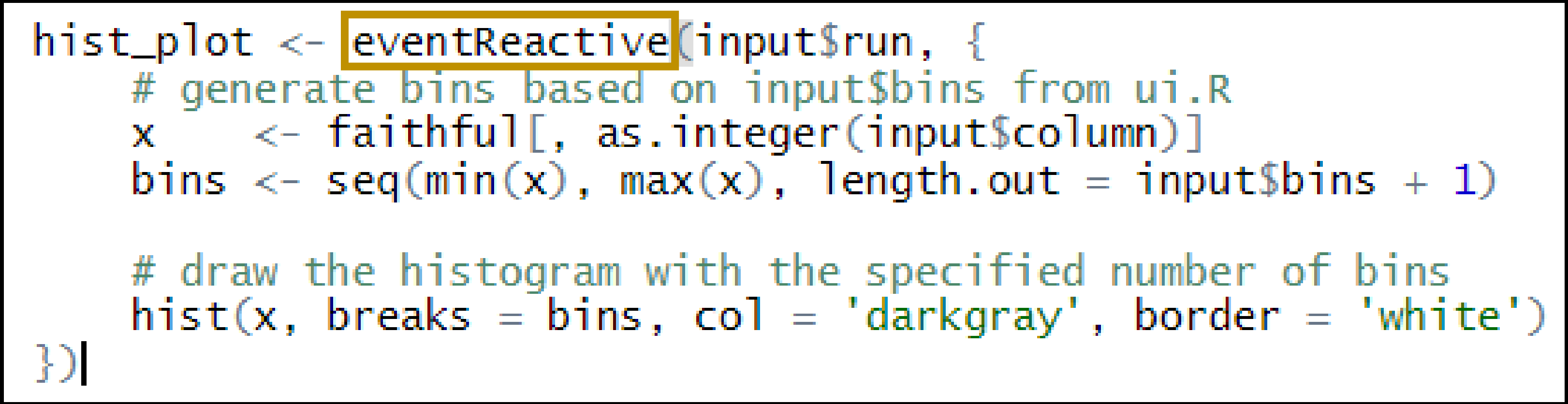
Eruption time

UI additions



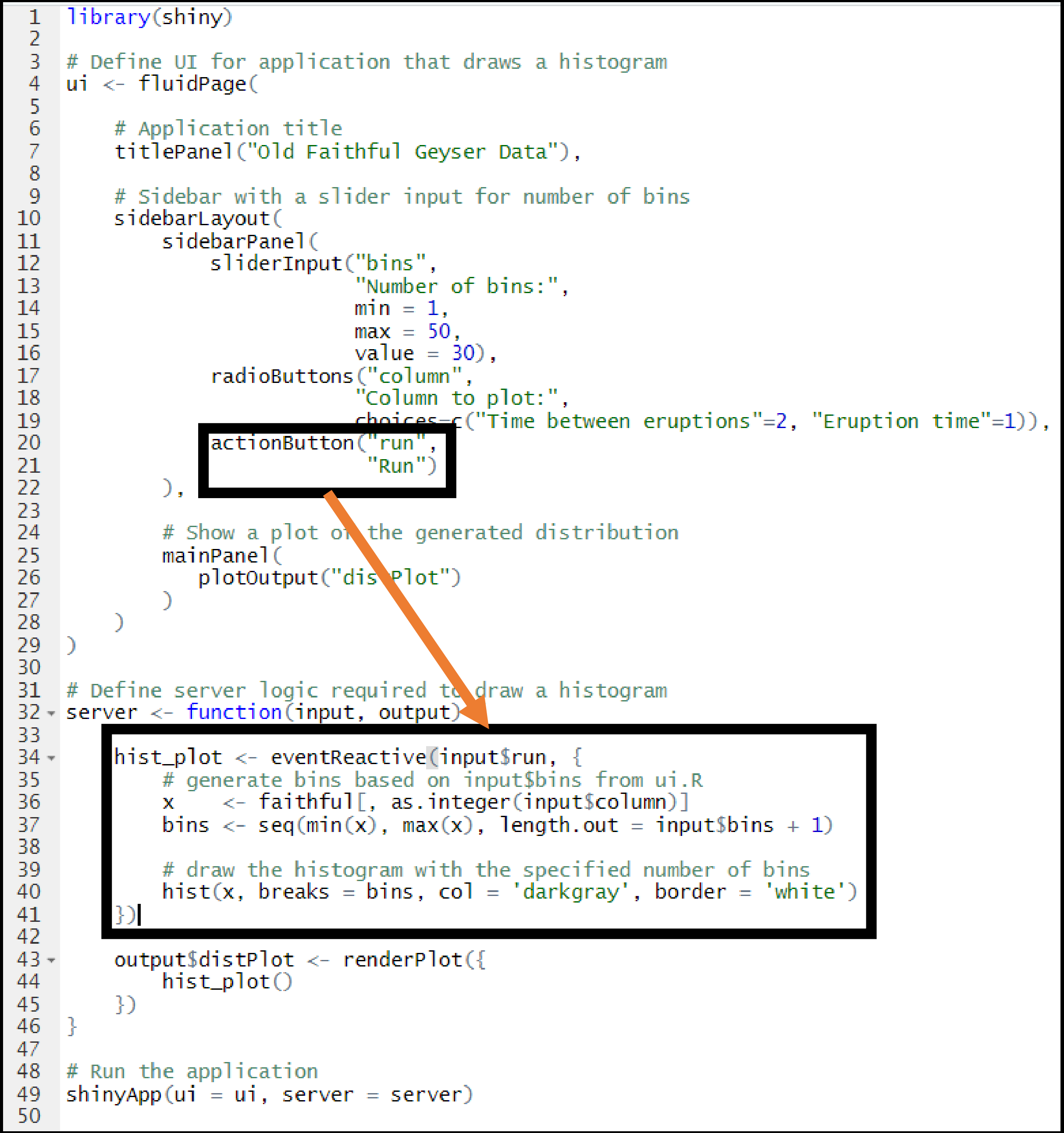
```
radioButtons("column",
             "Column to plot:",
             choices=c("Time between eruptions"=2, "Eruption time"=1)),
actionButton("run",
              "Run")
```

Server additions



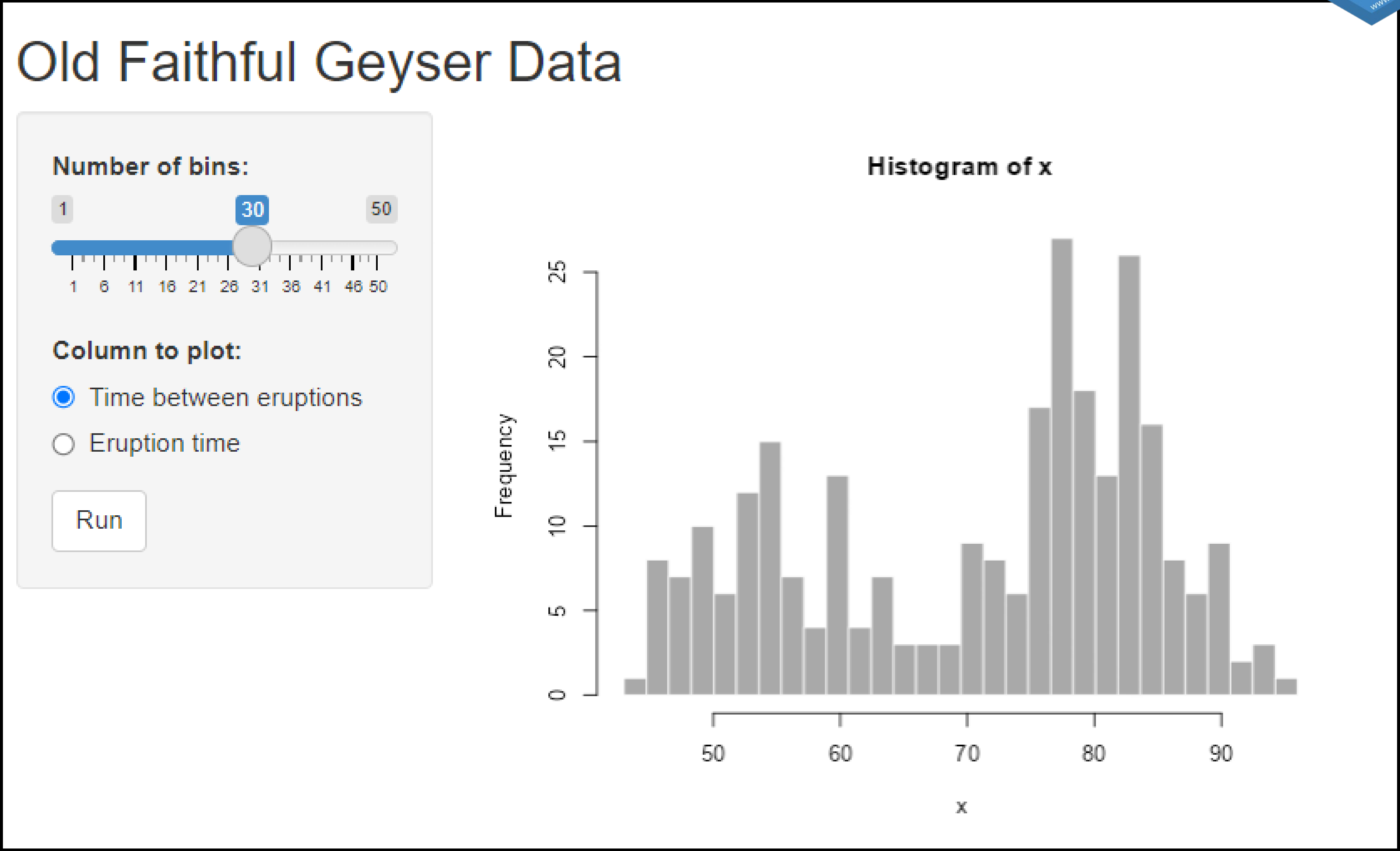
```
hist_plot <- eventReactive(input$run, {
    # generate bins based on input$bins from ui.R
    x    <- faithful[, as.integer(input$column)]
    bins <- seq(min(x), max(x), length.out = input$bins + 1)

    # draw the histogram with the specified number of bins
    hist(x, breaks = bins, col = 'darkgray', border = 'white')
})
```

More complex functionality



```r
library(shiny)

# Define UI for application that draws a histogram
ui <- fluidPage(

    # Application title
    titlePanel("Old Faithful Geyser Data"),

    # Sidebar with a slider input for number of bins
    sidebarLayout(
        sidebarPanel(
            sliderInput("bins",
                        "Number of bins:",
                        min = 1,
                        max = 50,
                        value = 30),
            radioButtons("column",
                        "Column to plot:",
                        choices=c("Time between eruptions"=2, "Eruption time"=1)),
            actionButton("run",
                         "Run")
        ),

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

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

    hist_plot <- eventReactive(input$run, {
        # generate bins based on input$bins from ui.R
        x    <- faithful[, as.integer(input$column)]
        bins <- seq(min(x), max(x), length.out = input$bins + 1)

        # draw the histogram with the specified number of bins
        hist(x, breaks = bins, col = 'darkgray', border = 'white')
    })

    output$distPlot <- renderPlot({
        hist_plot()
    })
}

# Run the application
shinyApp(ui = ui, server = server)
```

More complex app


Old Faithful Geyser Data

Number of bins:

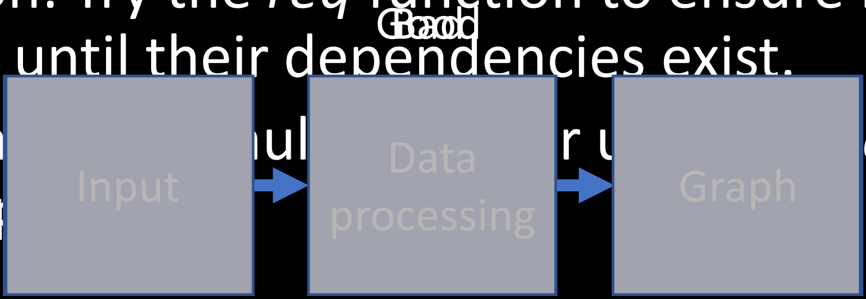
Column to plot:

[✓ Time between eruptions]

Eruption time

Debugging

- Shiny apps are hard to debug
- Start by running code outside app.
- Many bugs I have been unable to replicate outside the app have been related to an improper sequence of events.
  - Solution: Try the *req* function to ensure no processes trigger until their dependencies exist.
  - Also ca ul r u *else* to control the ap



Additional packages to improve your app

- **ggiraph** or **plotly**: Hoverable interactive graphs
- **shinywidgets**: More and prettier inputs
- **leaflet**: interactive maps
- **leaflet.minicharts**: pie and bar charts over maps

- Can also customize app by inserting HTML or CSS

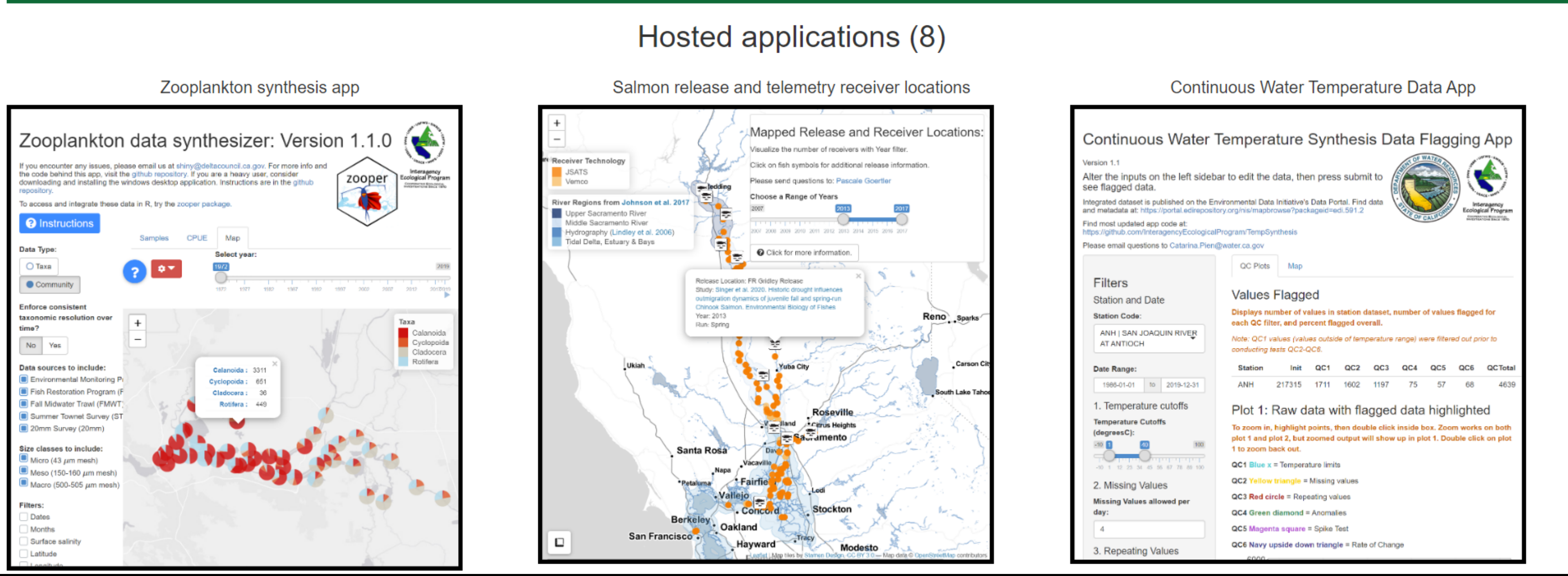
+
−
Taxa
Calanoida
Cyclopoida
Mysida
Cladocera
Rotifera

Deploying your app

- Online at shinyapps.io
  - Free or paid account tiers
  - The Delta Science program has a paid account and may be willing to host your app
- Or host it online on your own server if you have the time, skill, and money.

Example shiny apps from the Delta

- https://baydeltalive.com/fish/hatchery-releases
- https://deltascience.shinyapps.io/home (links to all DSP hosted apps)



Hosted applications (8)

Zooplankton synthesis app

Zooplankton data synthesizer: Version 1.1.0

If you encounter any issues, please email us at shiny@deltacouncil.ca.gov. For more info and the code behind this app, visit the github repository. If you are a heavy user, consider downloading and installing the windows desktop application. Instructions are in the github repository.

To access and integrate these data in R, try the zooper package.

Data Type: Taxa [✓ Community]

Enforce consistent taxonomic resolution over time? [✓ No] Yes

Data sources to include:
- ✓ Environmental Monitoring P
- ✓ Fish Restoration Program (F
- ✓ Fall Midwater Trawl (FMWT
- ✓ Summer Townet Survey (ST
- ✓ 20mm Survey (20mm)

Size classes to include:
- ✓ Micro (43 μm mesh)
- ✓ Meso (150-160 μm mesh)
- ✓ Macro (500-505 μm mesh)

Filters:
- ☐ Dates
- ☐ Months
- ☐ Surface salinity
- ☐ Latitude

Select year: 1972

Calanoida :	3311
Cyclopoida :	651
Cladocera :	36
Rotifera :	449

Salmon release and telemetry receiver locations

Mapped Release and Receiver Locations:

Visualize the number of receivers with Year filter.

Click on fish symbols for additional release information.

Please send questions to: Pascale Goertler

Choose a Range of Years: 2013 – 2017

Click for more information.

Release Location: FR Gridley Release
Study: Singer et al. 2020. Historic drought influences outmigration dynamics of juvenile fall and spring-run Chinook Salmon. Environmental Biology of Fishes
Year: 2013
Run: Spring

Continuous Water Temperature Data App

Continuous Water Temperature Synthesis Data Flagging App

Version 1.1

Alter the inputs on the left sidebar to edit the data, then press submit to see flagged data.

Integrated dataset is published on the Environmental Data Initiative's Data Portal. Find data and metadata at: https://portal.edirepository.org/nis/mapbrowse?packageid=edi.591.2

Find most updated app code at: https://github.com/InteragencyEcologicalProgram/TempSynthesis

Please email questions to Catarina.Pien@water.ca.gov

Filters

Station and Date

Station Code: ANH | SAN JOAQUIN RIVER AT ANTIOCH

Date Range: 1986-01-01 to 2019-12-31

1. Temperature cutoffs

Temperature Cutoffs (degreesC): 1 – 40

2. Missing Values

Missing Values allowed per day: 4

3. Repeating Values

Values Flagged

Displays number of values in station dataset, number of values flagged for each QC filter, and percent flagged overall.

Note: QC1 values (values outside of temperature range) were filtered out prior to conducting tests QC2-QC6.

Station	Init	QC1	QC2	QC3	QC4	QC5	QC6	QCTotal
ANH	217315	1711	1602	1197	75	57	68	4639

Plot 1: Raw data with flagged data highlighted

To zoom in, highlight points, then double click inside box. Zoom works on both plot 1 and plot 2, but zoomed output will show up in plot 1. Double click on plot 1 to zoom back out.

QC1 Blue x = Temperature limits

QC2 Yellow triangle = Missing values

QC3 Red circle = Repeating values

QC4 Green diamond = Anomalies

QC5 Magenta square = Spike Test

QC6 Navy upside down triangle = Rate of Change

QC Plots
Map
Interagency Ecological Program
Department of Water Resources State of California

Resources for learning

- https://shiny.posit.co/r/articles/
- https://shiny.posit.co/r/getstarted/shiny-basics/lesson1/index.html
- https://shiny.posit.co/r/gallery/

- Google