Inputs

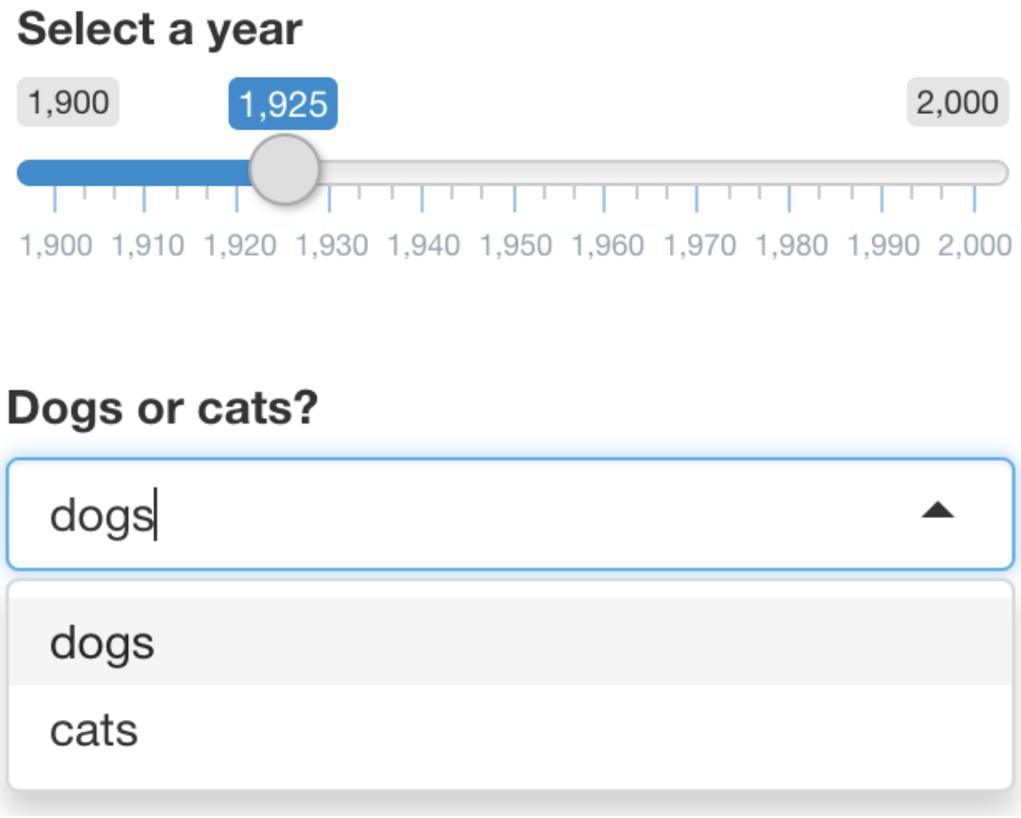
BUILDING WEB APPLICATIONS WITH SHINY IN R

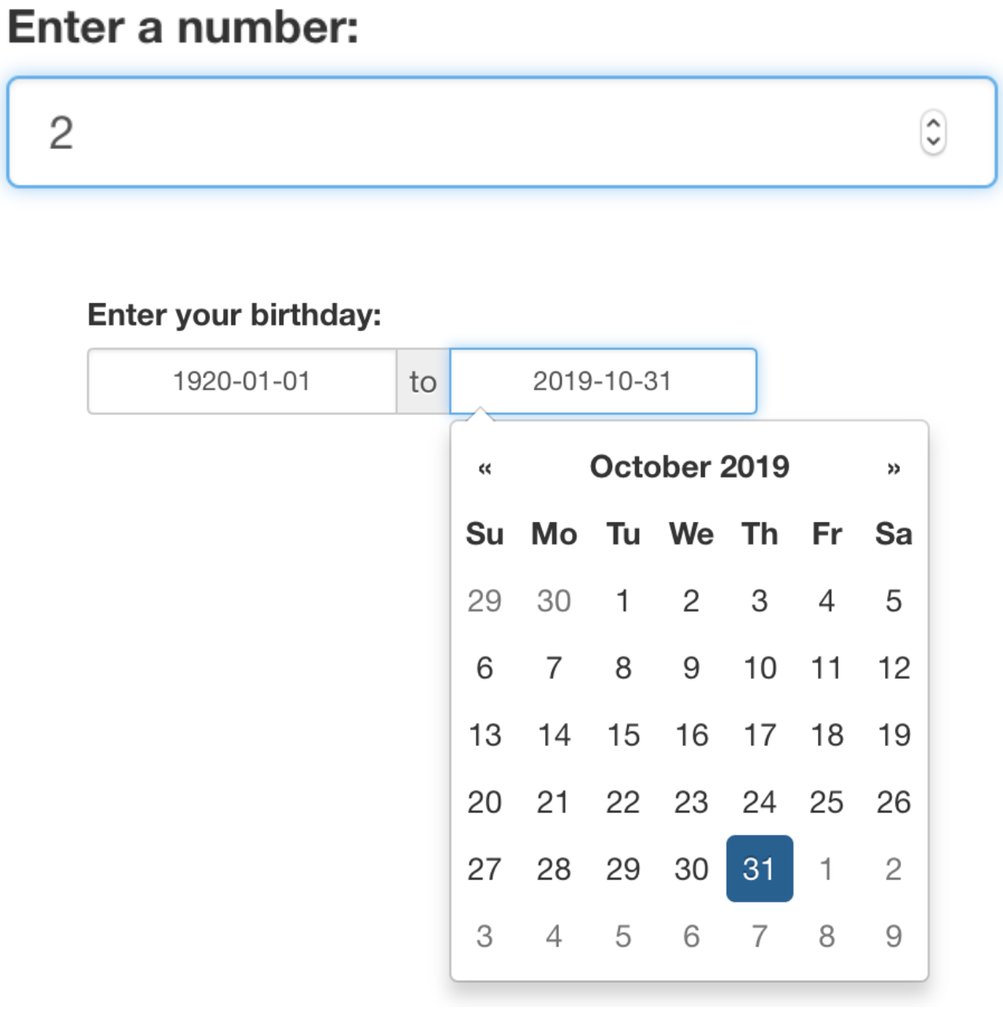
Kaelen Medeiros
Data Scientist

datacamp

Example inputs

Shiny provides a variety of inputs to choose from.





Input functions

```
selectInput("inputId",
            "label",
            choices = c("A", "B", "C"))
```

```
sliderInput("inputId",
            "label",
            value = 1925,
            min = 1900,
            max = 2000)
```

```
?dateRangeInput
help(checkboxInput)
```

Where to use inputs

```
ui <- fluidPage(
  textInput("name", "Enter a name:"),
  selectInput("animal", "Dogs or cats?", choices = c("dogs", "cats")),
  textOutput("greeting"),
  textOutput("answer")
)
server <- function(input, output, session) {
  output$greeting <- renderText({
    paste("Do you prefer dogs or cats,", input$name, "?")
  })
  output$answer <- renderText({
    paste("I prefer", input$animal, "!")
  })
}
```

Let's practice!

BUILDING WEB APPLICATIONS WITH SHINY IN R

Outputs

BUILDING WEB APPLICATIONS WITH SHINY IN R

Kaelen Medeiros
Data Scientist

Render functions

```
ui <- fluidPage(
  textInput("name", "Enter a name:"),
  selectInput("animal", "Dogs or cats?", choices = c("dogs", "cats")),
  textOutput("question"),
  textOutput("answer")
)

server <- function(input, output, session) {
  output$question <- renderText({
    paste("Do you prefer dogs or cats,", input$name, "?")
  })
  output$answer <- renderText({
    paste("I prefer", input$animal, "!")
  })
}
```

Other render functions

- `renderTable()`
- `renderImage()`
- `renderPlot()`
- **Shiny documentation**

Output functions

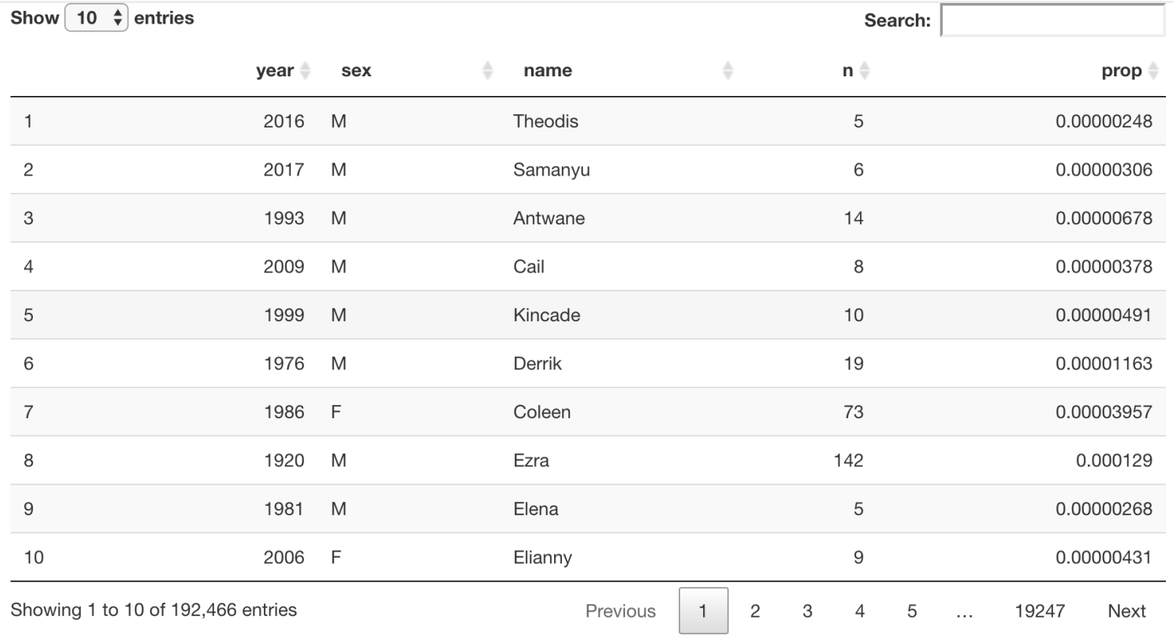
```
ui <- fluidPage(
  textInput("name", "Enter a name:"),
  selectInput("animal", "Dogs or cats?", choices = c("dogs", "cats")),
  textOutput("question"),
  textOutput("answer")
)
```

Other output functions

- `tableOutput()` or `dataTableOutput`
- `imageOutput()`
- `plotOutput()`

Non-Shiny output and render functions

```
library(shiny)
library(babynames)

ui <- fluidPage(
  DT::DTOutput("babynames_table")
)

server <- function(input, output){
  output$babynames_table <- DT::renderDT({
    babynames %>%
      dplyr::slice_sample(prop = .1) %>%
      DT::datatable()
  })
}

shinyApp(ui = ui, server = server)
```



Show 10 entries Search:

	year	sex	name	n	prop
1	2016	M	Theodis	5	0.00000248
2	2017	M	Samanyu	6	0.00000306
3	1993	M	Antwane	14	0.00000678
4	2009	M	Cail	8	0.00000378
5	1999	M	Kincade	10	0.00000491
6	1976	M	Derrik	19	0.00001163
7	1986	F	Coleen	73	0.00003957
8	1920	M	Ezra	142	0.000129
9	1981	M	Elena	5	0.00000268
10	2006	F	Elianny	9	0.00000431

Showing 1 to 10 of 192,466 entries Previous 1 2 3 4 5 … 19247 Next

Let's practice!

BUILDING WEB APPLICATIONS WITH SHINY IN R

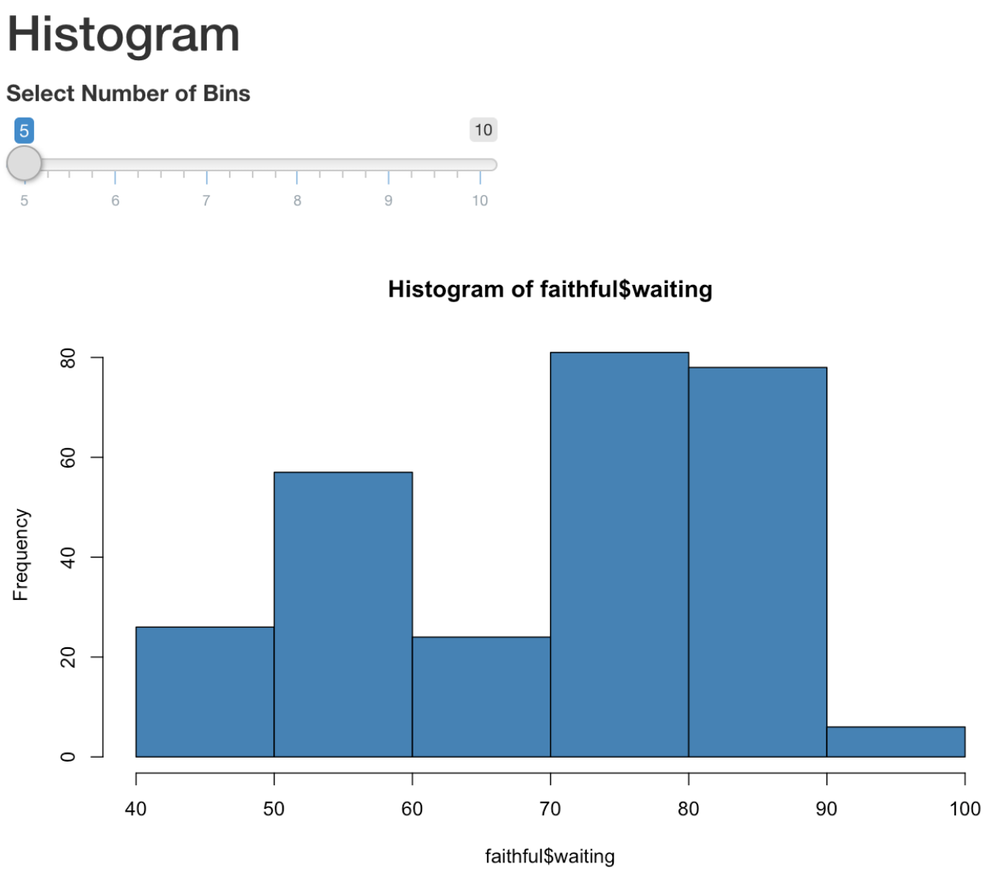
Layouts and themes

BUILDING WEB APPLICATIONS WITH SHINY IN R

Kaelen Medeiros
Data Scientist

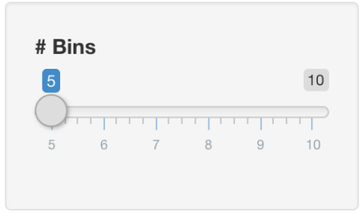
Default Shiny app layout

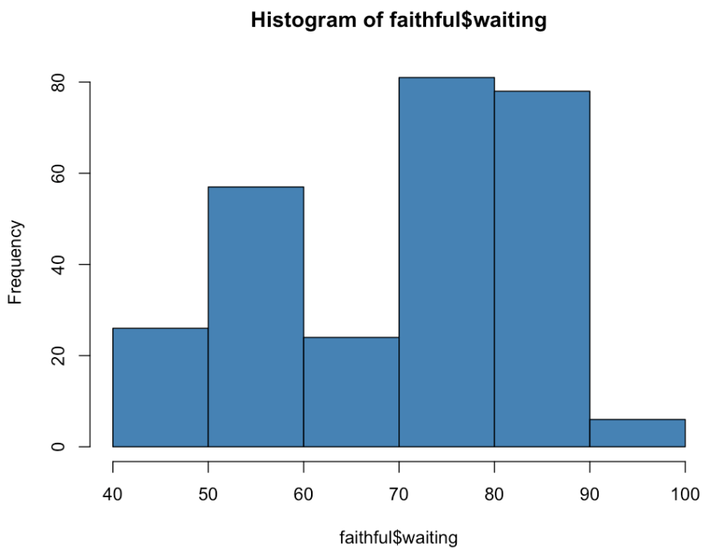
```
ui <- fluidPage(
  titlePanel("Histogram"),
  sliderInput('nb_bins', '# Bins', 5, 10, 5),
  plotOutput('hist')
)
server <- function(input, output, session){
  output$hist <- renderPlot({
    hist(faithful$waiting,
         breaks = input$nb_bins,
         col = 'steelblue')
  })
}
shinyApp(ui = ui, server = server)
```



Sidebar layout

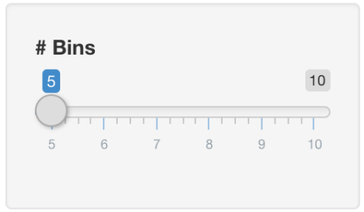
```
ui <- fluidPage(
  titlePanel("Histogram"),
  sidebarLayout(
    sidebarPanel(sliderInput('nb_bins',
                             '# Bins', 5, 10, 5)),
    mainPanel(plotOutput('hist'))
  )
)
server <- function(input, output, session){
  output$hist <- renderPlot({
    hist(faithful$waiting, breaks = input$nb_bins,
         col = 'steelblue')
  })
}
shinyApp(ui = ui, server = server)
```

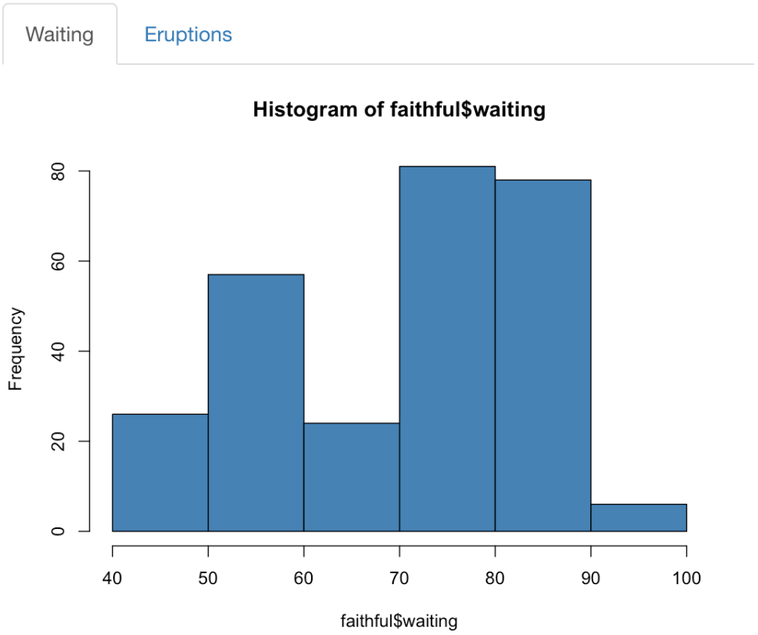




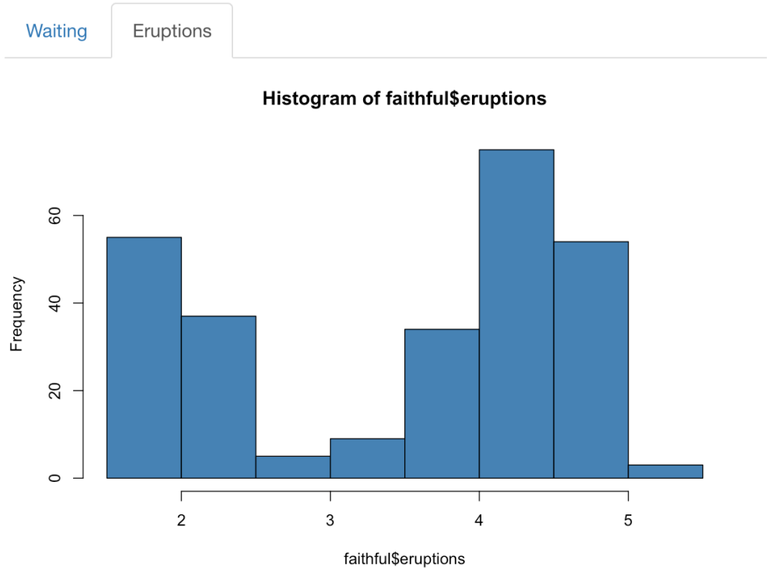
Tab layout

Histogram





Histogram



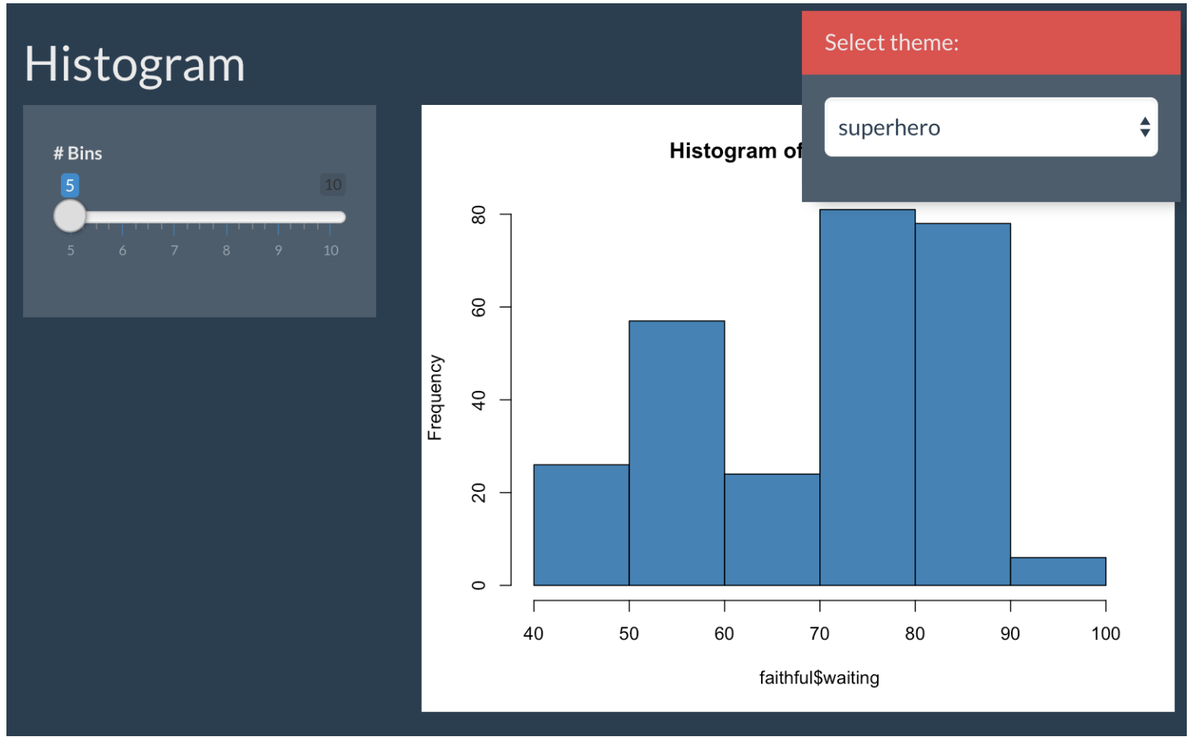
Tab layout

```
ui <- fluidPage(
  titlePanel("Histogram"),
  sidebarLayout(
    sidebarPanel(sliderInput('nb_bins', '# Bins',
                             5, 10, 5)),
    mainPanel(
      tabsetPanel(
        tabPanel('Waiting',
                 plotOutput('hist_waiting')),
        tabPanel('Eruptions',
                 plotOutput('hist_eruptions'))
      )
    )
  )
)
```

```
server <- function(input, output, session){
  output$hist_waiting <- renderPlot({
    hist(faithful$waiting,
         breaks = input$nb_bins,
         col = 'steelblue')
  })
  output$hist_eruptions <- renderPlot({
    hist(faithful$eruptions,
         breaks = input$nb_bins,
         col = 'steelblue')
  })
}
shinyApp(ui = ui, server = server)
```

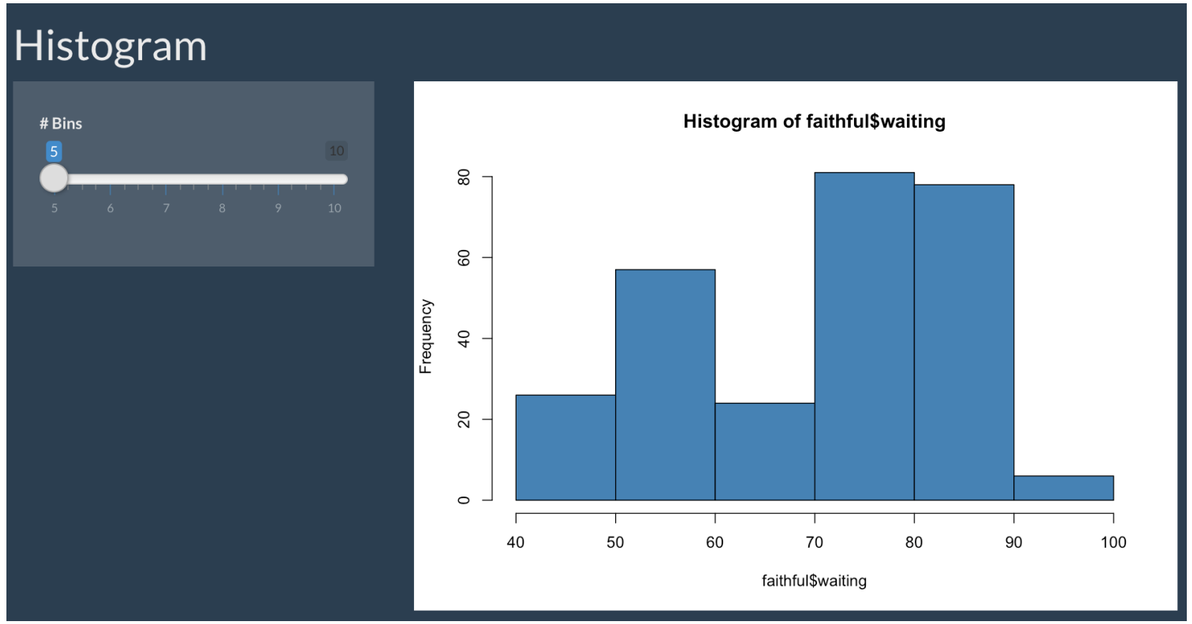
Theme selector

```
ui <- fluidPage(
  titlePanel("Histogram"),
  shinythemes::themeSelector(),
  sidebarLayout(
    sidebarPanel(sliderInput('nb_bins', '# Bins',
                             5, 10, 5)),
    mainPanel(plotOutput('hist'))
  )
)
server <- function(input, output, session){
  output$hist <- renderPlot({
    hist(faithful$waiting, breaks = input$nb_bins,
         col = 'steelblue')
  })
}
shinyApp(ui = ui, server = server)
```



Adding a theme

```
ui <- fluidPage(
  titlePanel("Histogram"),
  theme = shinythemes::shinytheme('superhero'),
  sidebarLayout(
    sidebarPanel(sliderInput('nb_bins', '# Bins',
                             5, 10, 5)),
    mainPanel(plotOutput('hist'))
  )
)
server <- function(input, output, session){
  output$hist <- renderPlot({
    hist(faithful$waiting, breaks = input$nb_bins,
         col = 'steelblue')
  })
}
shinyApp(ui = ui, server = server)
```



Let's practice!

BUILDING WEB APPLICATIONS WITH SHINY IN R

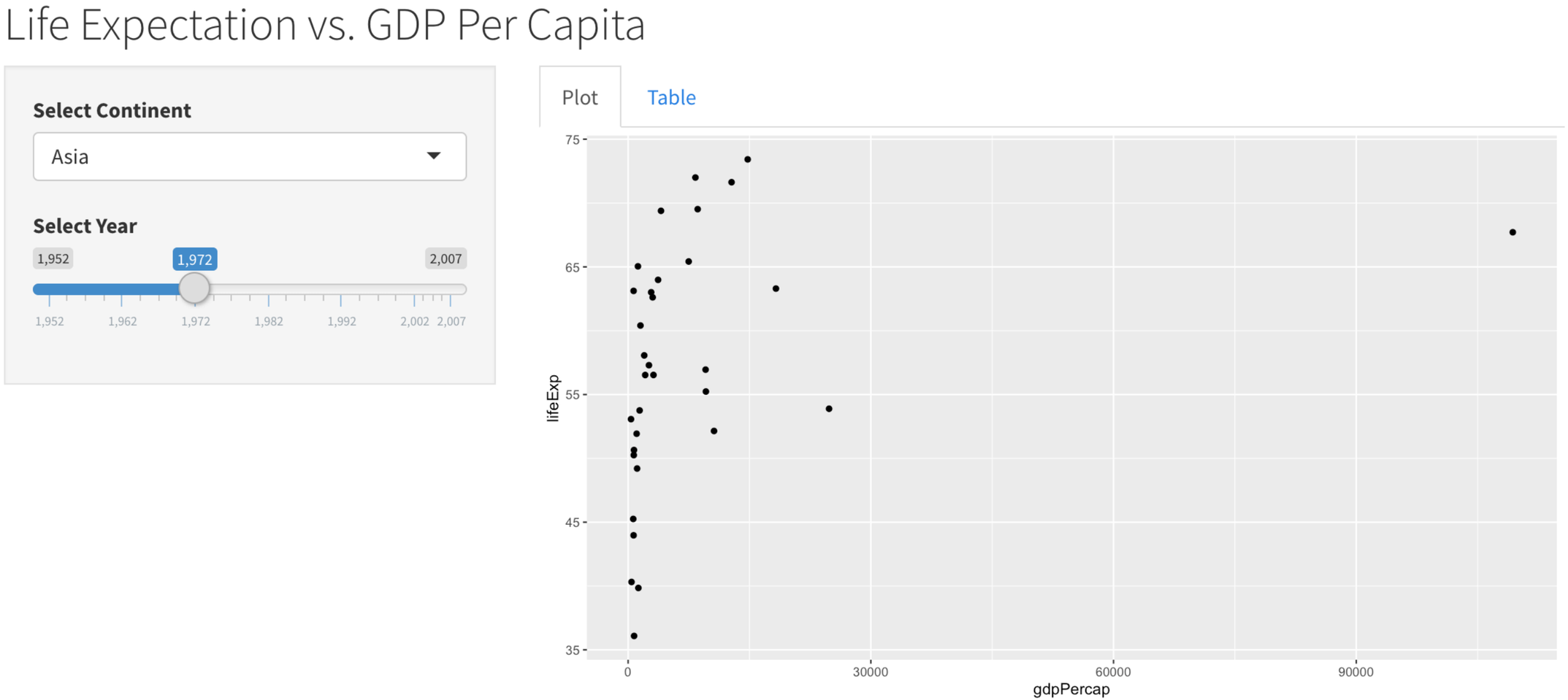
Building apps

BUILDING WEB APPLICATIONS WITH SHINY IN R

Kaelen Medeiros
Data Scientist

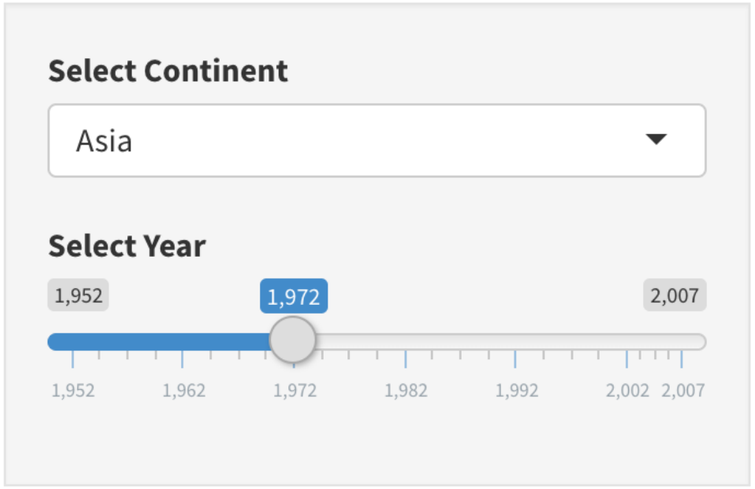
datacamp

Explore Life Expectation vs. GDP per Capita



Explore Life Expectation vs. GDP per Capita

Life Expectation vs. GDP Per Capita



Plot | Table

Show 10 entries Search:

	country	continent	year	lifeExp	pop	gdpPercap
1	Afghanistan	Asia	1972	36.088	13079460	739.9811058
2	Bahrain	Asia	1972	63.3	230800	18268.65839
3	Bangladesh	Asia	1972	45.252	70759295	630.2336265
4	Cambodia	Asia	1972	40.317	7450606	421.6240257
5	China	Asia	1972	63.11888	862030000	676.9000921
6	Hong Kong, China	Asia	1972	72	4115700	8315.928145
7	India	Asia	1972	50.651	567000000	724.032527
8	Indonesia	Asia	1972	49.203	121282000	1111.107907

Building Shiny apps: 4 steps

1. Add inputs (UI)
2. Add outputs (UI/Server)
3. Update layout (UI)
4. Update outputs (Server)

Step 1: Add inputs (UI)

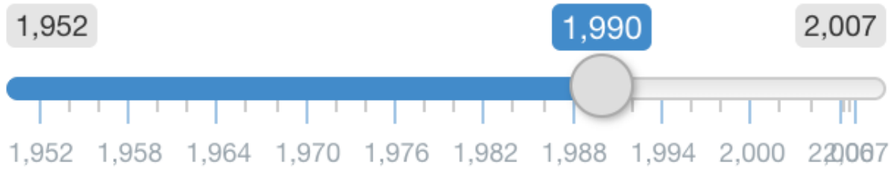
```
ui <- fluidPage(
  titlePanel("Life Expectation vs. GDP Per Capita"),
  selectInput('continent', 'Select Continent', unique(gapminder$continent)),
  sliderInput('year', 'Select Year', 1952, 2007, 1990, step = 5)
)
server <- function(input, output, session){


}
shinyApp(ui = ui, server = server)
```

Life Expectation vs. GDP Per Capita

Select Continent

Asia

Select Year



Step 2: Add outputs (UI)

```
ui <- fluidPage(
  titlePanel("Life Expectation vs. GDP Per Capita"),
  selectInput('continent', 'Select Continent', unique(gapminder$continent)),
  sliderInput('year', 'Select Year', 1952, 2007, 1990, step = 5),
  plotOutput('plot'),
  DT::DTOutput('table')
)
```

Step 2: Add outputs (Server)

```
server <- function(input, output, session){
  output$plot <- renderPlot({
    ggplot()
  })
  output$table <- DT::renderDT({
    gapminder
  })
}
```

Life Expectation vs. GDP Per Capita

Select Continent

Asia

Select Year

1,952 1,990 2,007

1,952 1,958 1,964 1,970 1,976 1,982 1,988 1,994 2,000 2,007

Show 10 **entries**

	country	continent	year	lifeExp
1	Afghanistan	Asia	1952	28.801
2	Afghanistan	Asia	1957	30.332

Step 3: Update layout (UI)

```
ui <- fluidPage(
  titlePanel("Life Expectation vs. GDP Per Capita"),
  sidebarLayout(
    sidebarPanel(
      selectInput('continent', 'Select Continent', unique(gapminder$continent)),
      sliderInput('year', 'Select Year', 1952, 2007, 1990, step = 5)
    ),
    mainPanel(
      plotOutput('plot'),
      DT::DTOutput('table')
    )
  )
)
```

Step 3: Update layout (UI)

```
ui <- fluidPage(
  titlePanel("Life Expectation vs. GDP Per Capita"),
  sidebarLayout(
    sidebarPanel(
      selectInput('continent', 'Select Continent', unique(gapminder$continent)),
      sliderInput('year', 'Select Year', 1952, 2007, 1990, step = 5)
    ),
    mainPanel(
      tabsetPanel(
        tabPanel("Plot", plotOutput('plot')),
        tabPanel("Table", DT::DTOutput('table'))
      )
    )
  )
)
```

Life Expectation vs. GDP Per Capita

Select Continent

Asia

Select Year

1,952 1,990 2,007

1,952 1,964 1,976 1,988 2,000 2,007

Plot | Table

Show 10 entries Search:

	country	continent	year	lifeExp	pop	gdpPercap
1	Afghanistan	Asia	1952	28.801	8425333	779.4453145
2	Afghanistan	Asia	1957	30.332	9240934	820.8530296
3	Afghanistan	Asia	1962	31.997	10267083	853.10071
4	Afghanistan	Asia	1967	34.02	11537966	836.1971382
5	Afghanistan	Asia	1972	36.088	13079460	739.9811058
6	Afghanistan	Asia	1977	38.438	14880372	786.11336
7	Afghanistan	Asia	1982	39.854	12881816	978.0114388
8	Afghanistan	Asia	1987	40.822	13867957	852.3959448

Step 4: Update outputs (Server)

```
server <- function(input, output, session){
  output$plot <- renderPlot({
    data <- gapminder %>%
      filter(year == input$year) %>%
      filter(continent == input$continent)
    print(data)
    ggplot(data, aes(x = gdpPercap, y = lifeExp)) +
      geom_point()
  })
  output$table <- DT::renderDT({
    gapminder %>%
      filter(year == input$year) %>%
      filter(continent == input$continent)
  })
}
```

Life Expectation vs. GDP Per Capita

Select Continent

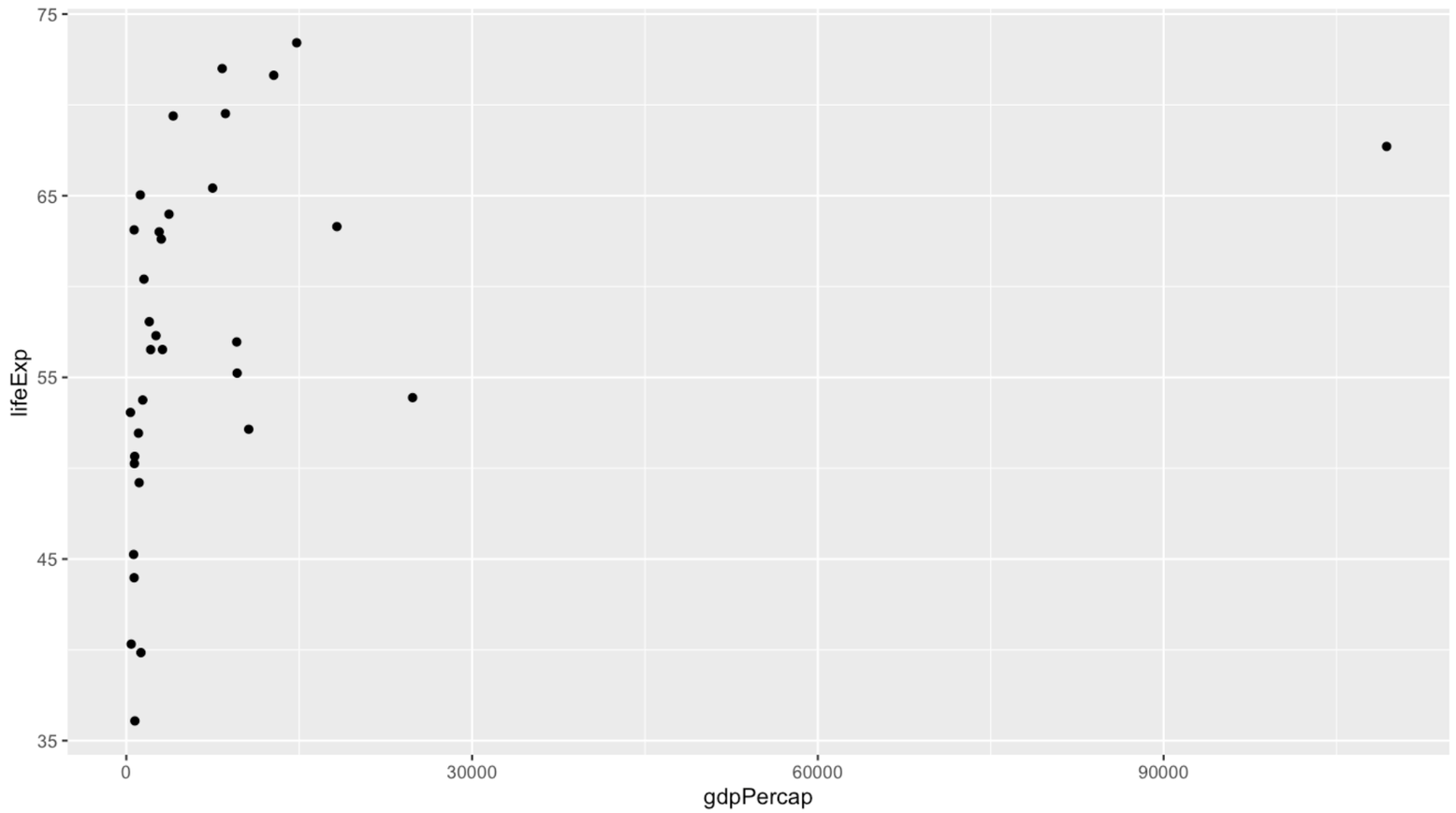
Asia

Select Year

1,952 1,972 2,007

1,952 1,962 1,972 1,982 1,992 2,002 2,007

Plot Table



Let's practice!

BUILDING WEB APPLICATIONS WITH SHINY IN R