

Building interactive visualizations with R

Pau Palop-García and Felix Haaß

GIGA German Institute of Global and Area Studies

June 2018

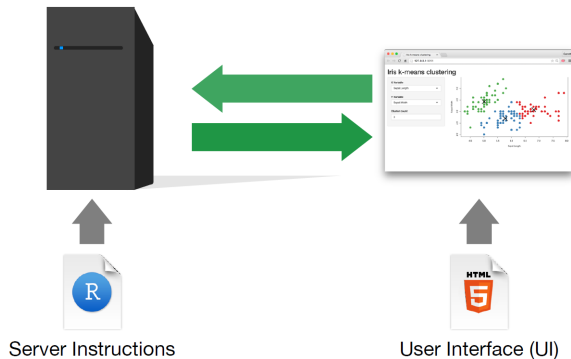
Introduction

- ▶ What is an interactive visualization?
- ▶ Course outline

Main steps for building an app

- ▶ Set main goal of the app
- ▶ Install R Studio and necessary packages (e.g. "shiny")
- ▶ Select visualization
- ▶ Prepare the data accordingly
- ▶ Code the app
- ▶ Deploy and share the app

An app...



© CC 2015 RStudio, Inc.

Selecting the appropriate visualization

Telephones by region

Region:

N.Amer

Data from AT&T (1961) The World's Telephones.



© CC 2015 RStudio, Inc.

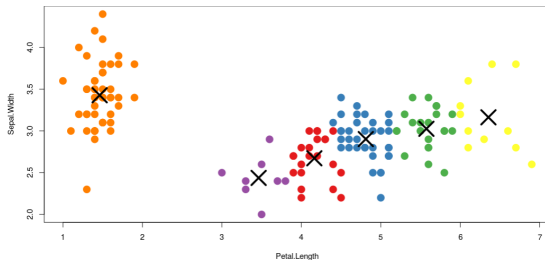
Selecting the appropriate visualization

Iris k-means clustering

X Variable
Petal.Length

Y Variable
Sepal.Width

Cluster count
6



© CC 2015 RStudio, Inc.

Selecting the appropriate visualization

Word Cloud

Choose a book:

A Mid Summer Night's Dream ▼

Change

Minimum Frequency:

1 15 50

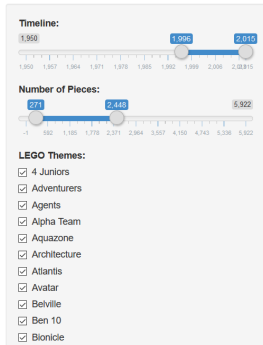
Maximum Number of Words:

1 100 300



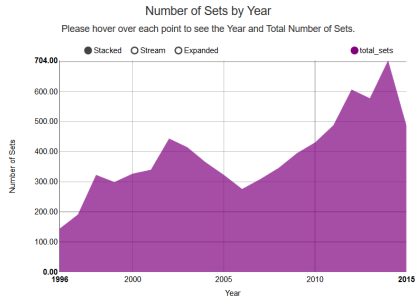
© CC 2015 RStudio, Inc.

Selecting the appropriate visualization



Dataset

Visualize the Data



© CC 2015 RStudio, Inc.

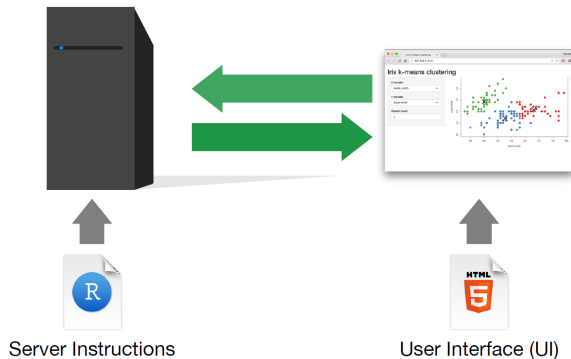
Preparing the data

- ▶ Depends on the app
- ▶ General rule: keep it simple

Structure of a ShinyApp

- ▶ **A user interface (ui):** controls the app layout
- ▶ **A server function (server):** contains all the functions needed to build the app
- ▶ **A call to the shinyApp function:** creates a shiny app that pairs ui and server

Building the simplest app



© CC 2015 RStudio, Inc.

Building the simplest app

```
library(shiny)

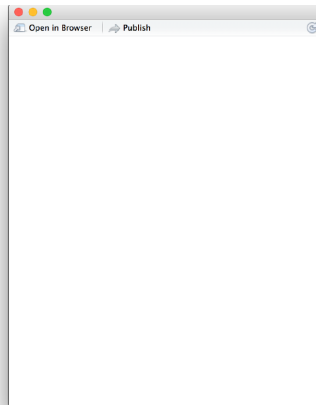
ui <- fluidPage(

)

server <- function(input, output) {

}

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



© CC 2015 RStudio, Inc.

Building the simplest app

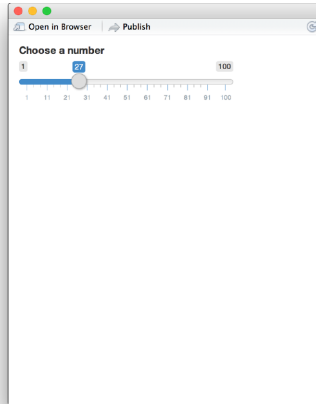
```
library(shiny)

ui <- fluidPage(
  sliderInput(inputId = "num",
    label = "Choose a number",
    value = 25, min = 1, max = 100)
)

server <- function(input, output) {

}

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



© 2015 RStudio, Inc.

© CC 2015 RStudio, Inc.

Widgets

- ▶ **Definition:** a web element that users can interact with
- ▶ A way for users to send messages to the Shiny app
- ▶ Widgets are R functions: they required at least two elements to work: a name and a label

Widgets

Buttons

Action

Submit

`actionButton()`
`submitButton()`

Single checkbox

☒ Choice A

`checkboxInput()`

Checkbox group

☒ Choice 1
☐ Choice 2
☐ Choice 3

`checkboxGroupInput()`

Date input

2014-01-01

`dateInput()`

Date range

2014-01-24 to 2014-01-24

`dateRangeInput()`

File input

Choose File No file chosen

`fileInput()`

Numeric input

1

`numericInput()`

Password Input

.....

`passwordInput()`

Radio buttons

☒ Choice 1
☐ Choice 2
☐ Choice 3

`radioButtons()`

Select box

Choice 1

`selectInput()`

Sliders



`sliderInput()`

Text input

Enter text...

`textInput()`

© CC 2015 RStudio, Inc.

© CC 2015 RStudio, Inc.

Building the simplest app

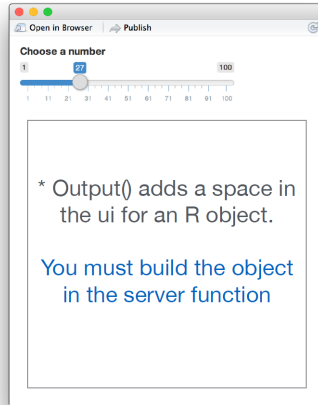
```
library(shiny)

ui <- fluidPage(
  sliderInput(inputId = "num",
    label = "Choose a number",
    value = 25, min = 1, max = 100),
  plotOutput("hist")
)

server <- function(input, output) {

}

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



Building the simplest app

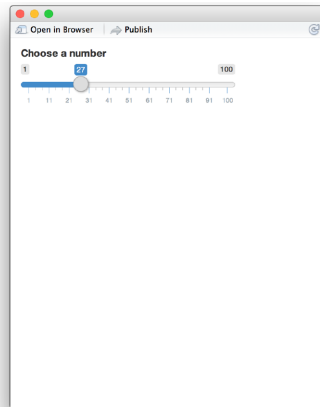
```
library(shiny)

ui <- fluidPage(
  sliderInput(inputId = "num",
    label = "Choose a number",
    value = 25, min = 1, max = 100),
  plotOutput("hist")
)

server <- function(input, output) {
  output$hist <-
}

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

1



Building the simplest app

```
library(shiny)

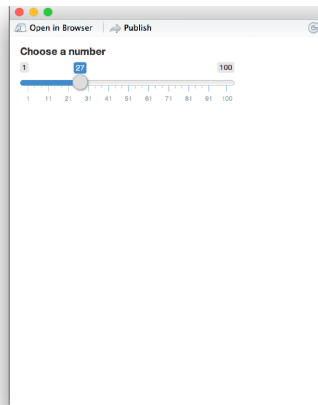
ui <- fluidPage(
  sliderInput(inputId = "num",
    label = "Choose a number",
    value = 25, min = 1, max = 100),
  plotOutput("hist")
)

server <- function(input, output) {
  output$hist <- renderPlot({

  })
}

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

2



Building the simplest app

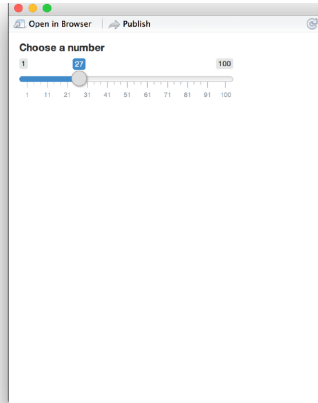
```
library(shiny)

ui <- fluidPage(
  sliderInput(inputId = "num",
    label = "Choose a number",
    value = 25, min = 1, max = 100),
  plotOutput("hist")
)

server <- function(input, output) {
  output$hist <- renderPlot({
    hist(rnorm(input$num))
  })
}

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

3



© CC 2015 RStudio, Inc.

Reactive outputs

Reactive outputs respond when users toggles a widget

Two main steps:

- ▶ Add an R object to the user interface
- ▶ Tell shiny how to build the object in the server function

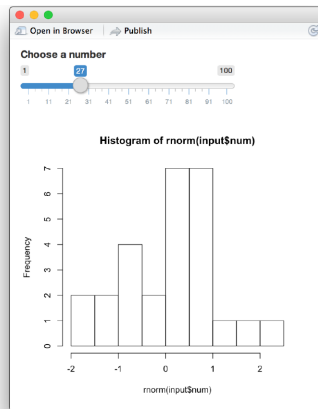
Building the simplest app

```
library(shiny)

ui <- fluidPage(
  sliderInput(inputId = "num",
    label = "Choose a number",
    value = 25, min = 1, max = 100),
  plotOutput("hist")
)

server <- function(input, output) {
  output$hist <- renderPlot({
    hist(rnorm(input$num))
  })
}

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



Deploying and sharing the app

Every shiny app is maintained by a computer running R How to save it:

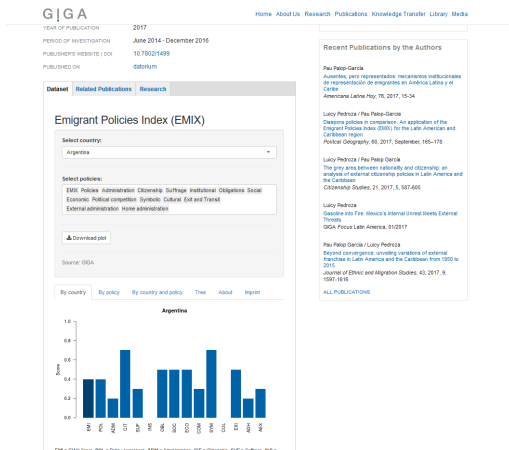
- ▶ app.R
- ▶ datasets, images, css, helper scripts, etc.

Deploying and sharing the app

Using Shinyapps.io

- ▶ Free
- ▶ Secure
- ▶ Scalable

Sharing in GIGA website via iframe



More info:

- ▶ <https://shiny.rstudio.com/tutorial/>
- ▶ <https://www.shinyapps.io/>
- ▶ <https://www.rstudio.com/>