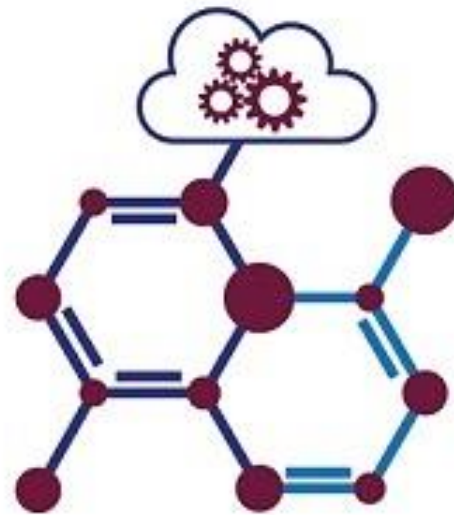




TECNOLÓGICO
NACIONAL DE MÉXICO®



CIENCIA DE
LOS DATOS

Diplomado en:

CIENCIA DE LOS DATOS E INTERNET DE LAS COSAS

VIRTUAL





MTI. Jose Gabriel Rodríguez Rivas, LI



MTI. Marzo Antonio Rodríguez Zúñiga, ISC



MAI. Rubén Pizarro Gurrola, LI

Objetivo

- Desarrollar aplicaciones R Shiny en R Studio
- Introducción

Temas

Temas	Estimado 20 hrs.
1. Reconocimiento de entorno de trabajo	1
2. Programación R	1
3. Estructuras de datos en R	2
4. Funciones del usuario en R	2
5. Visualización de datos	1
6. Importar y exportar datos	1
7. Librería <i>dplyr</i>	2
8. Archivos markdown y publicación rpubs	1
9. Simulación de datos	1
10. Gráficos avanzados <code>ggplot2()</code> y <code>plotty()</code>	2
11. Strings and Dates en R	1
12. Datos semiestructurados	2
13. Series de tiempo	1
14. R shiny	2

Pasado y presente



Pasado:

- HTML, CSS y JavaScript
- Análisis cuidadoso de los flujos de interacción para cuidar comportamientos de entrada y salida

Presente:

- R Shiny para el programador en R, significa hacer más fácil las aplicaciones WEB interactivas
- Crear UI encapsulando HTML, CSS y JavaScript
- Programación reactiva = “reaccionan” a los datos ejecutando una serie de eventos

¿Porqué R Shiny?

- Crear reportes (dashboard) que muestren indicadores y análisis de rendimiento
- Reemplazar documentos PDF y markdown haciéndolos interactivos
- Comunicar a una audiencia no tan técnica
- Análisis de datos de autoservicio, cargar datos y analizar dinámicamente
- Demostraciones interactivas y enseñanzas estadísticas
- Modificar entradas y observar resultados diferentes
- Promover y fomentar ciencia de los datos

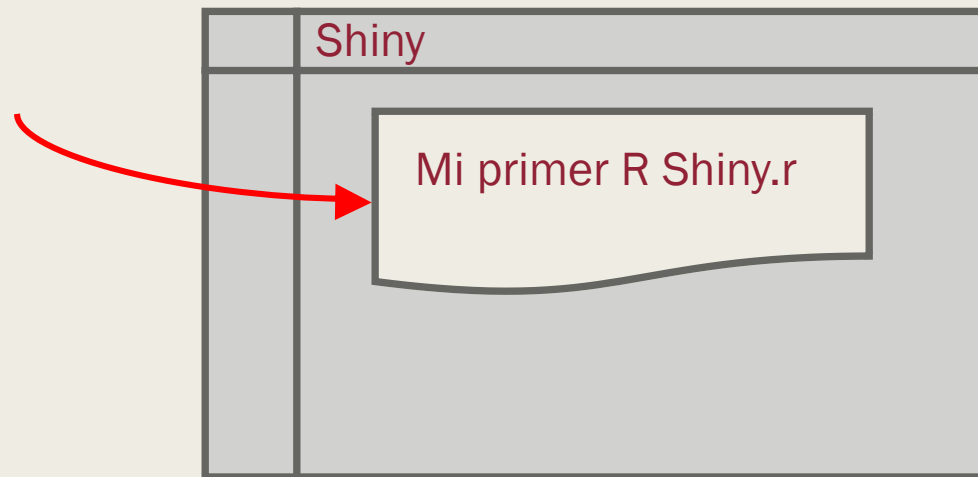


Paquetes R a instalar

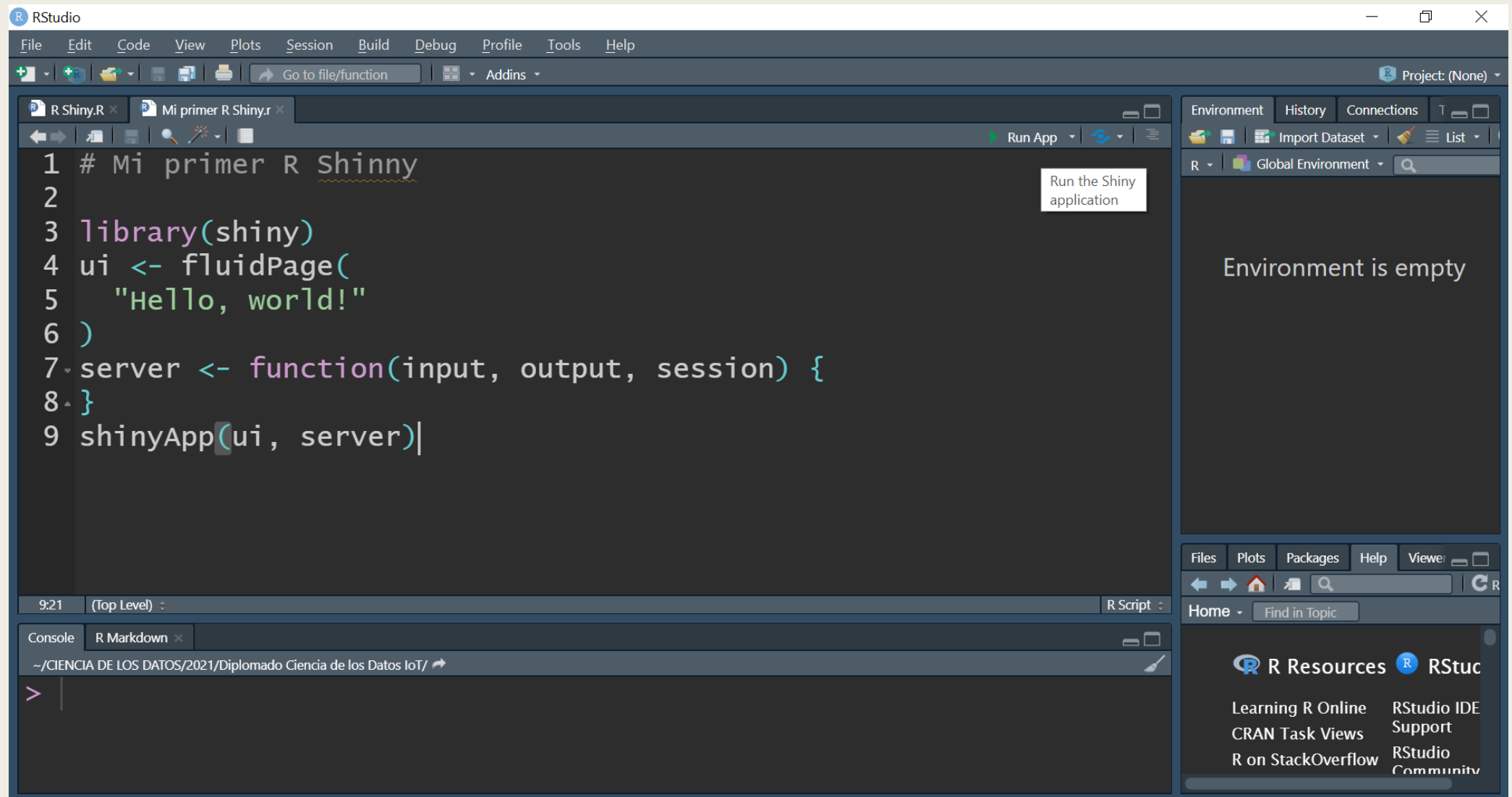
```
install.packages(c(  
  "gapminder", "ggforce", "gh", "globals", "openintro", "profvis",  
  "RSQLite", "shiny", "shinycssloaders", "shinyFeedback",  
  "shinythemes", "testthat", "thematic", "tidyverse", "vroom",  
  "waiter", "xml2", "zeallot"  
))
```


Primer R Shiny

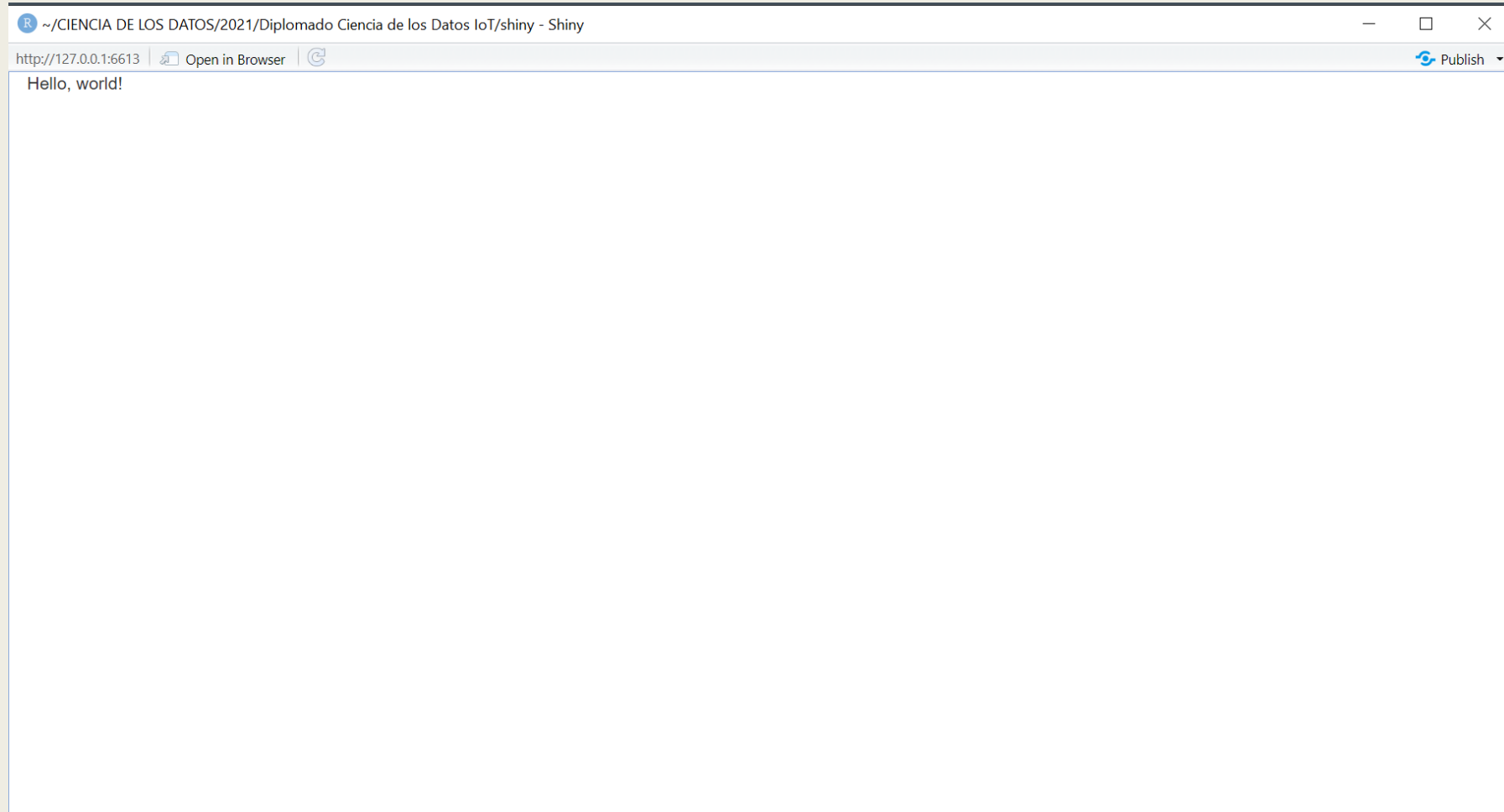
```
library(shiny)
ui <- fluidPage(
  "Hello, world!"
)
server <- function(input, output, session) {
}
shinyApp(ui, server)
```



Primer R Shiny

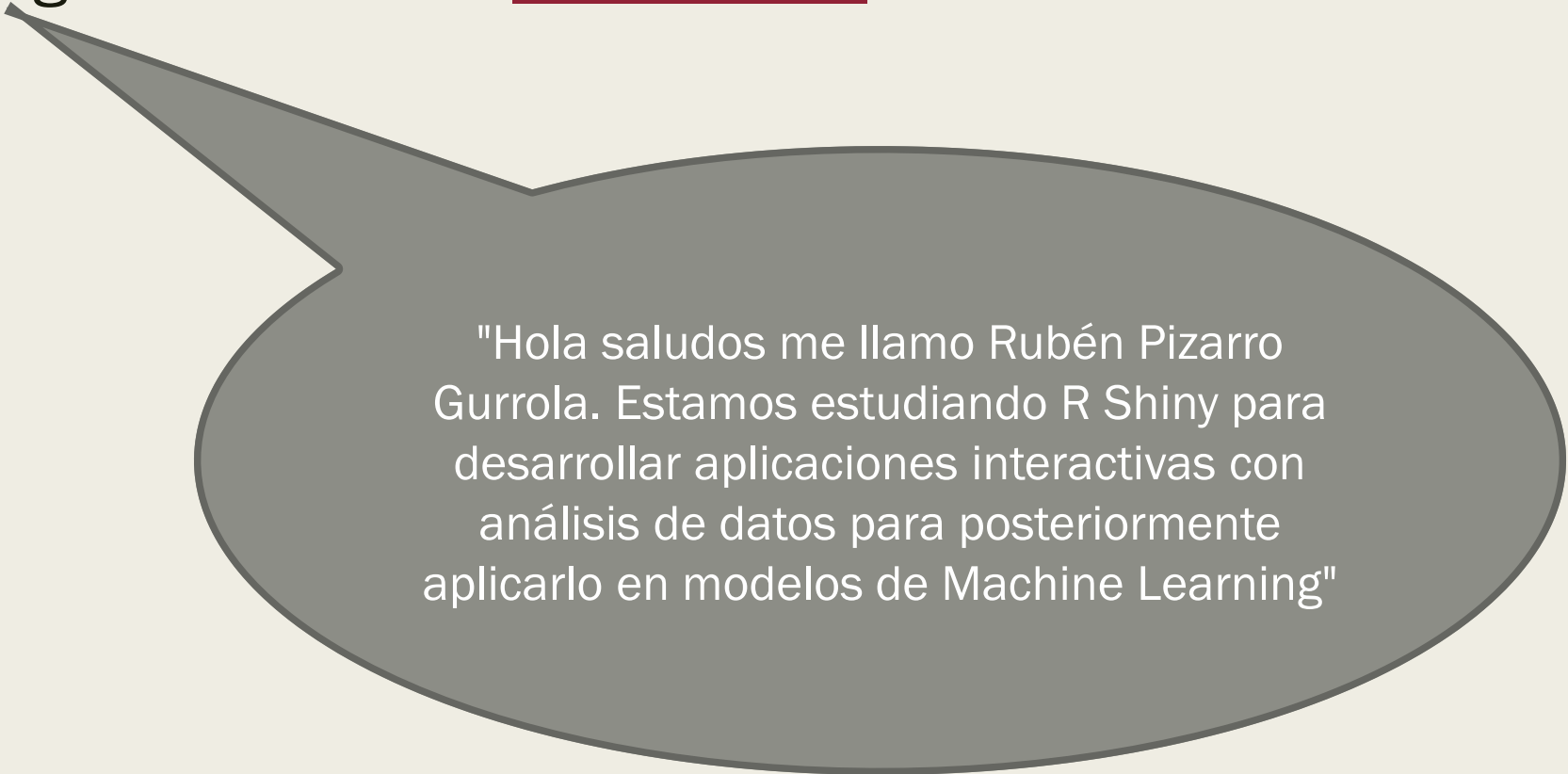


Primer R Shiny



Desafío 1

- Crear una aplicación R Shiny que muestre algún mensaje como este y guardarlo como 01.desafio.r:



"Hola saludos me llamo Rubén Pizarro Gurrola. Estamos estudiando R Shiny para desarrollar aplicaciones interactivas con análisis de datos para posteriormente aplicarlo en modelos de Machine Learning"

Agregando controles

```
# Agregando controles.r
```

```
library(shiny)
```

```
ui <- fluidPage(  
  
```

```
    selectInput(inputId = "dataset", label = "Dataset", choices =  
ls("package:datasets")),  
    
```

```
    verbatimTextOutput(outputId = "summary"),  
    
```

```
    tableOutput(outputId = "table")  
  )  
  server <- function(input, output, session) {  
  }  
  shinyApp(ui, server)
```

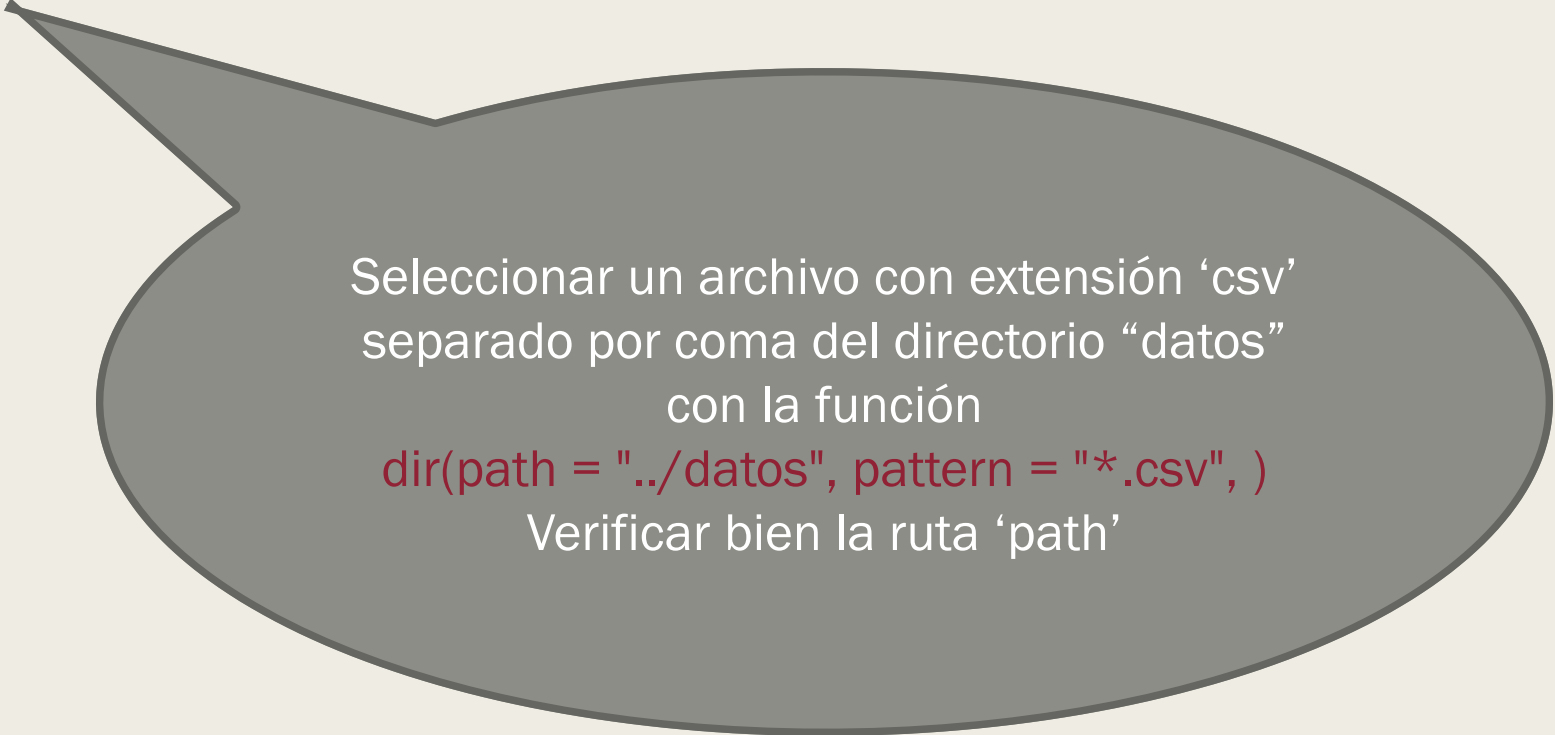
Función de Diseño

Selector de opciones <select>

verbatimTextOutput() y tableOutput() son
controles de salida

Desafío 2

- Crear una aplicación R Shiny permita seleccionar un archivo del directorio actual o de un directorio deseado [02.desafio.r](#):



Seleccionar un archivo con extensión 'csv'
separado por coma del directorio "datos"
con la función

```
dir(path = "../datos", pattern = "*.csv", )
```

Verificar bien la ruta 'path'

Comportamiento de salida (output)

```
server <- function(input, output, session) {
```

```
  output$summary <- renderPrint({
```

```
    datos <- get(x = input$dataset)
```

```
    summary(datos)
```

```
  })
```

Función para mostrar datos en forma de texto

`get()` Recupera el valor de la entrada dataset aunque se pudiera recuperar directamente

```
  output$table <- renderTable({
```

```
    datos <- get(x = input$dataset)
```

```
    datos
```

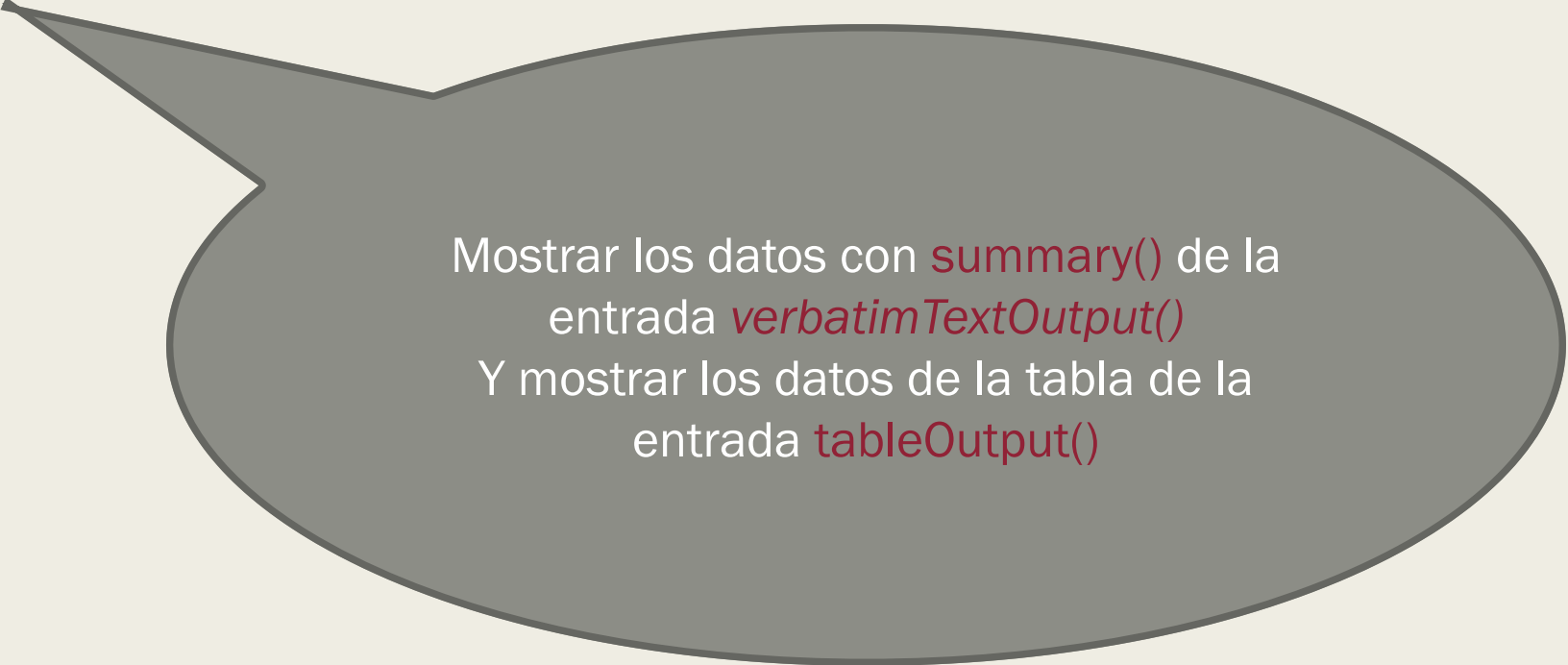
```
  })
```

```
}
```

Función para mostrar datos en forma de tabla

Desafío 3

- Complementar el desafío 2 incorporando la función de summary y mostrando los datos del archivo seleccionado con *dir()*.
- *Los archivos deben ser separados por como CSV*
- *La aplicación debe llamarse 03.desafio.r:*



Mostrar los datos con `summary()` de la
entrada `verbatimTextOutput()`
Y mostrar los datos de la tabla de la
entrada `tableOutput()`

Desafío 4

- Crear un selector con nombres de frutas y mostrar en pantalla el nombre de la fruta seleccionada 04.desafio.r:
- Agregar que sean a lo más 10 frutas.

Crear un selector que permita agregar
opciones de frutas y mostrar la opción
seleccionada

`c("Naranja", "Durazno", "Manzana", "Pera",
"Limón", "Guayaba", "Melón"),`

`"Carambola", "Yuzu", "Kiwi", "Pomelo"`

Programación reactiva


```
server <- function(input, output,
session) {
  output$summary <- renderPrint({
    datos <- get(x = input$dataset)
    summary(datos)
  })

  output$table <- renderTable({
    datos <- get(x = input$dataset)
    datos
  })
}
```

Función reactiva es que solo se cargue una vez y no varias veces y se pueda reutilizar `datos()` ya no se carga ni en la función `renderPrint()` ni en `renderTable()`

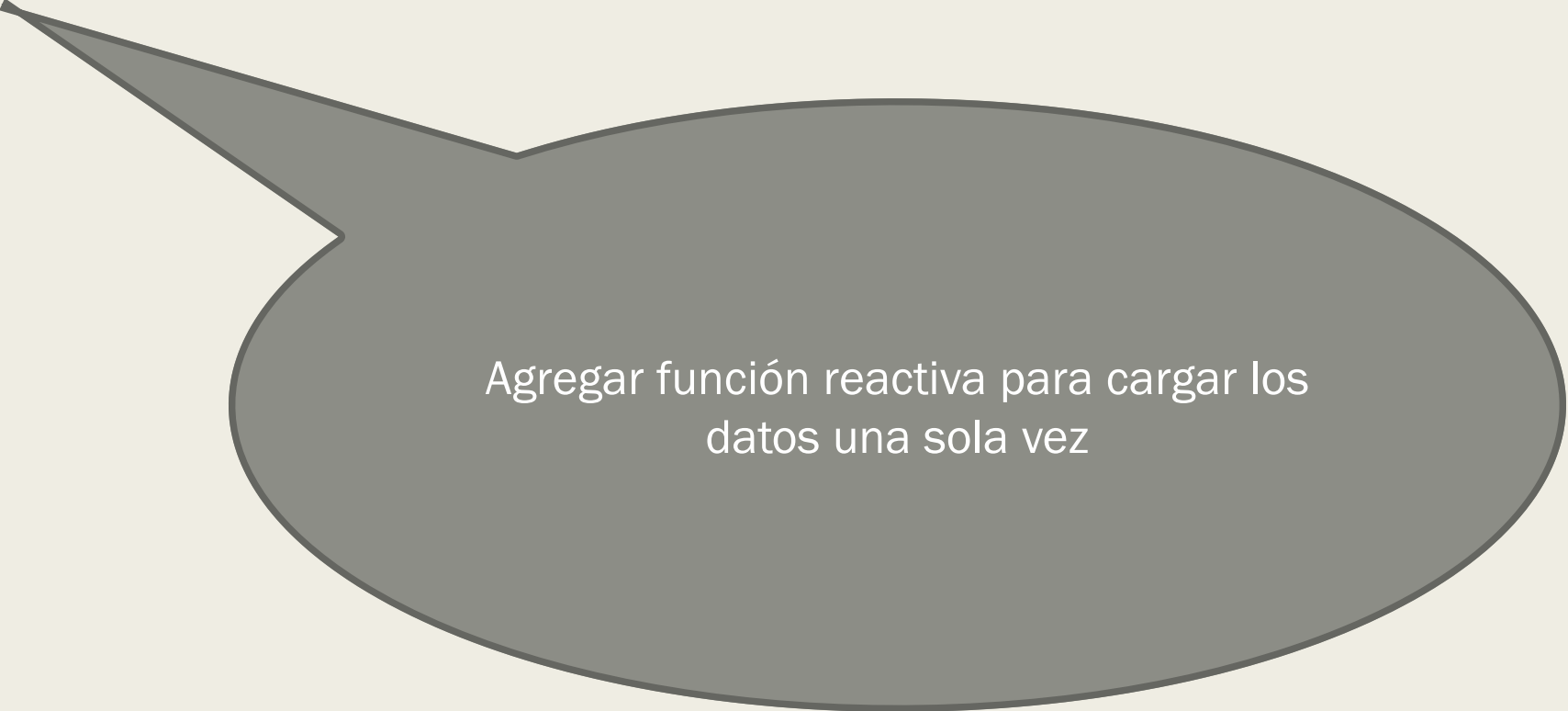
```
server <- function(input, output, session) {
  datos <- reactive({
    get(input$dataset,
"package:datasets")
  })
  output$summary <- renderPrint({
    summary(datos())
  })

  output$table <- renderTable({
    datos()
  })
}
```



Desafío 5

- Implementar una función reactiva llamada `datos()` a partir del desafío 3 y llamar al archivo [05.desafio.r](#):



Agregar función reactiva para cargar los
datos una sola vez

Otros controles de entrada tipo texto

textInput() Permite capturar una cadena de caracteres

textInput(inputId, label, value = "", width = NULL, placeholder = NULL)

numericInput() Permite capturar un valor numérico dentro de un rango establecido

*numericInput(inputId, label, value, min = NA, max = NA,
step = NA, width = NULL
)*

Otros controles de entrada tipo texto

Controles de entrada texto

```
library(shiny)
```

```
ui <- fluidPage(
```

```
  textInput(inputId = "nombre", label = "Nombres:"),
```

```
  numericInput(inputId = "edad", label = "Edad:", min = 18, max = 65, step = 1, value =  
  round(mean(18:65),2)),
```

```
  textOutput("saludos")
```

```
)
```

```
server <- function(input, output, session) {
```

```
  output$saludos <- renderPrint({
```

```
    saludos <- paste("Hola", input$nombre, "tienes ", input$edad, "años" )
```

```
    saludos
```

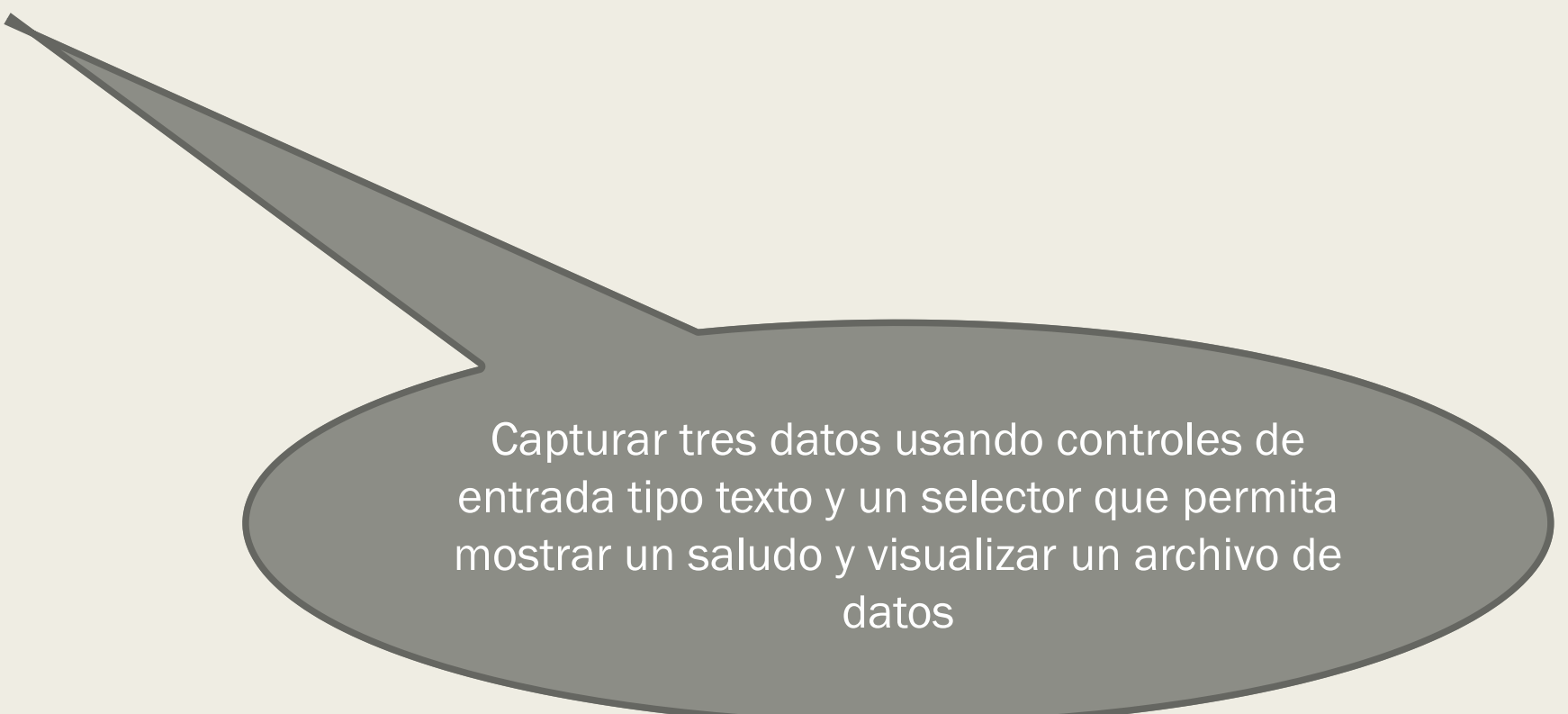
```
  })
```

```
}
```

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

Desafío 6

- Crear una aplicación shiny que permita capturar un nombre, una edad y seleccionar un estado de la república y mostrar un saludo, al igual que mostrar un archivo de datos, un data.frame [06.desafio.r](#):
- El archivo de datos se encuentra en

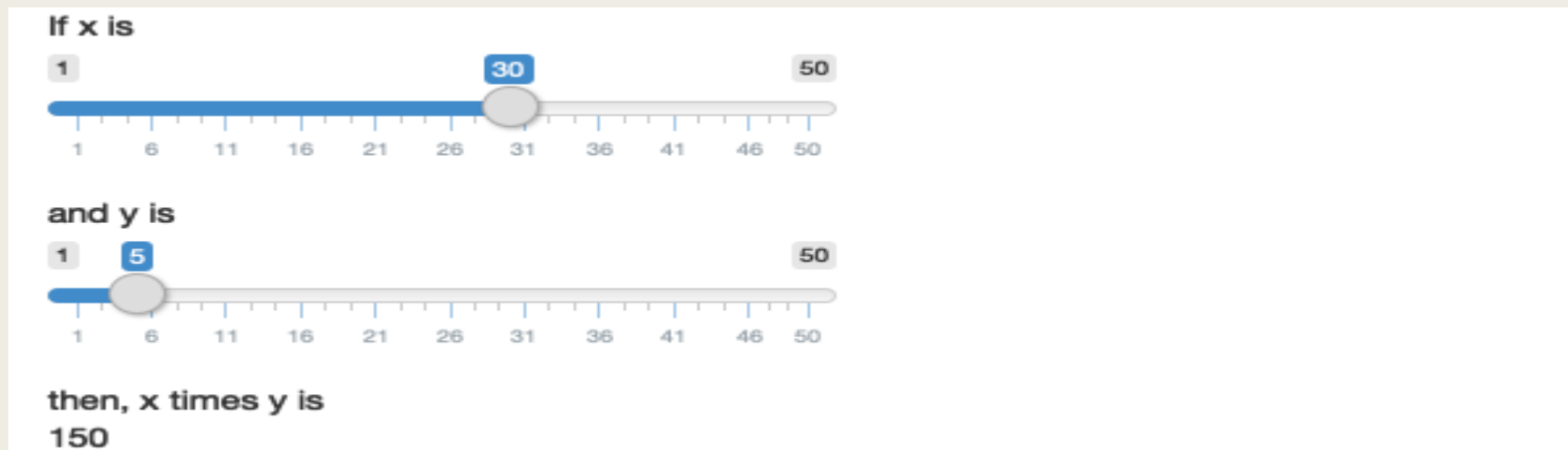


Capturar tres datos usando controles de entrada tipo texto y un selector que permita mostrar un saludo y visualizar un archivo de datos

Control *sliderInput()*

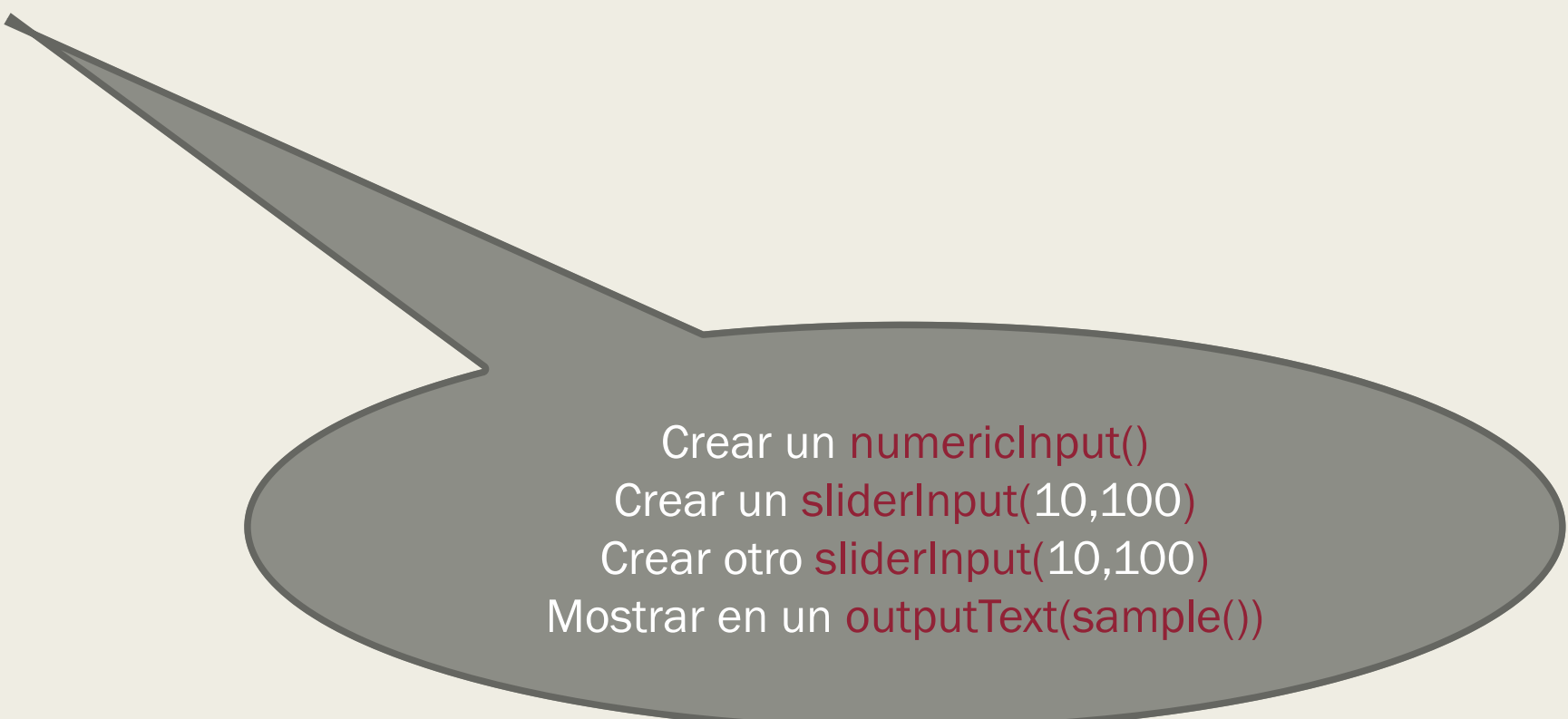
- `sliderInput()` . Permite dar entrada a un dato numérico por medio de un slider

```
sliderInput( inputId, label, min, max, value, step = NULL,  
round = FALSE,  
format = NULL, locale = NULL, ticks = TRUE,  animate = FALSE,  width  
= NULL, sep = ",", pre = NULL, post = NULL,  
timeFormat = NULL,  timezone = NULL,  dragRange = TRUE  
)
```



Desafío 7

- Crear una aplicación shiny que muestre dos `slider()` y una entrada numérica `numericInput()` y muestre a manera textual o con tabla una muestra de n números definido por la entrada numérica con valores de un rango entre el valor del primer slider y el valor del segundo slider [07.desafio.r](#)



```
Crear un numericInput()  
Crear un sliderInput(10,100)  
Crear otro sliderInput(10,100)  
Mostrar en un outputText(sample())
```

Bibliografía

- Bibliografía
- Wickhan, H. (2020). Mastering Shiny. (C. C.-N.-N. License., Ed.) USA: O'Reilly Media. Obtenido [de https://mastering-shiny.org/basic-ui.html](https://mastering-shiny.org/basic-ui.html)