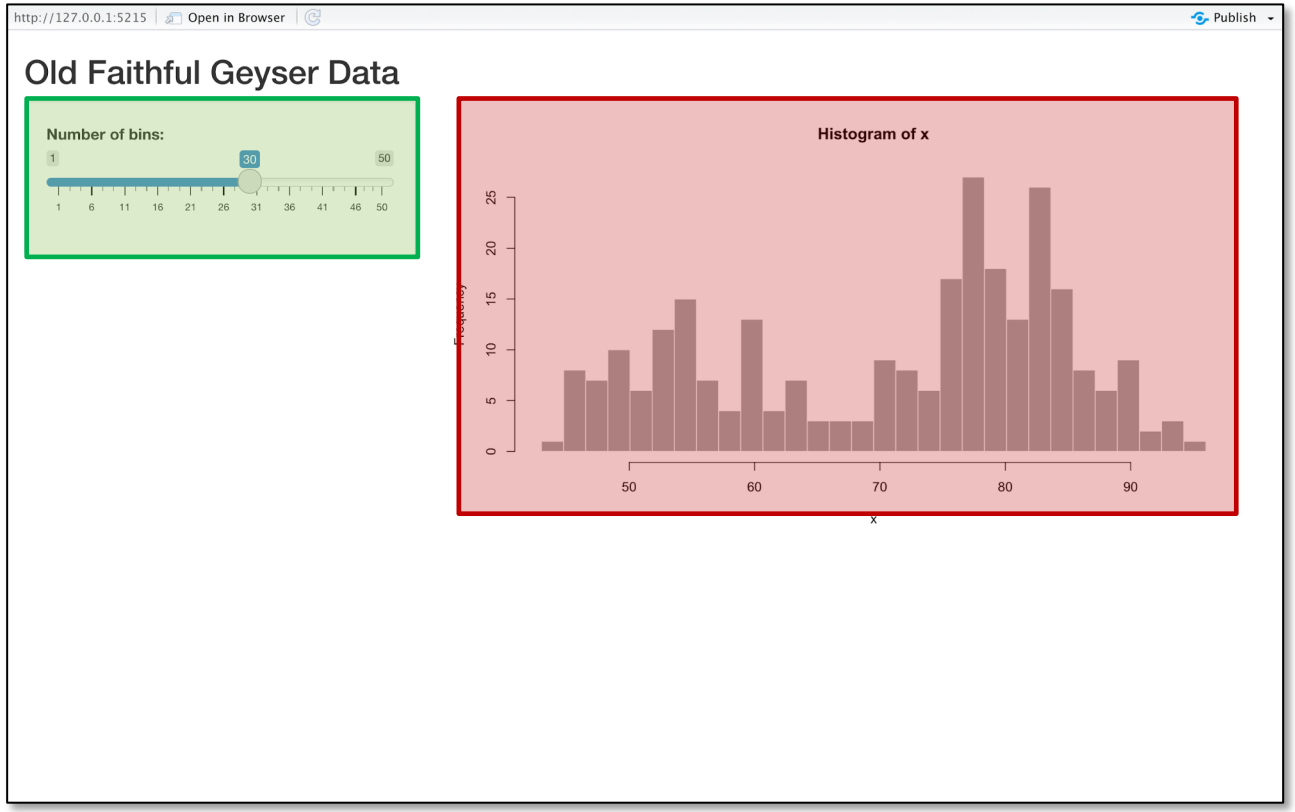
TEMA 2

INPUT OUTPUT SERVER

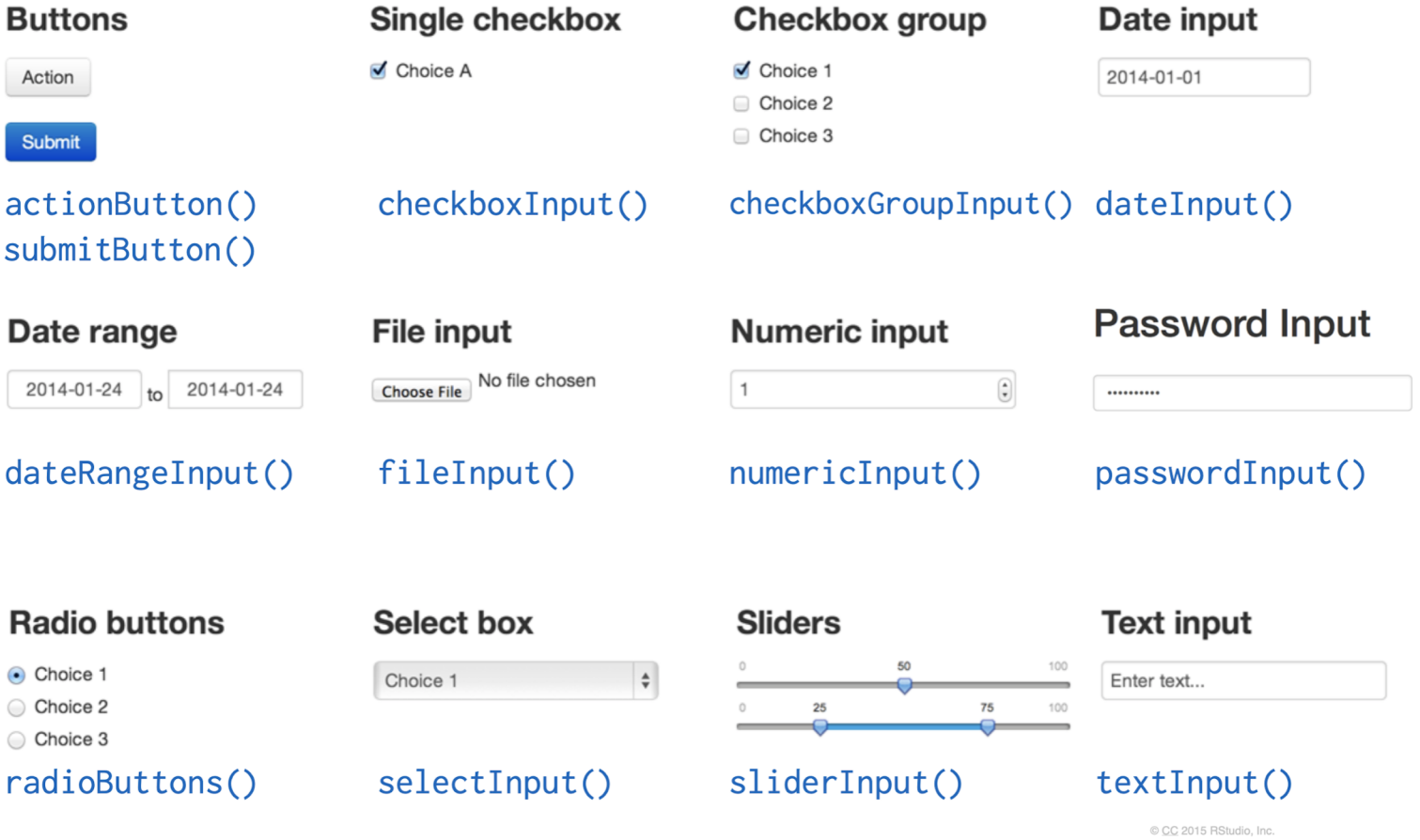
CONSTRUYE TU APP

La app se construye agregando elementos como inputs y outputs



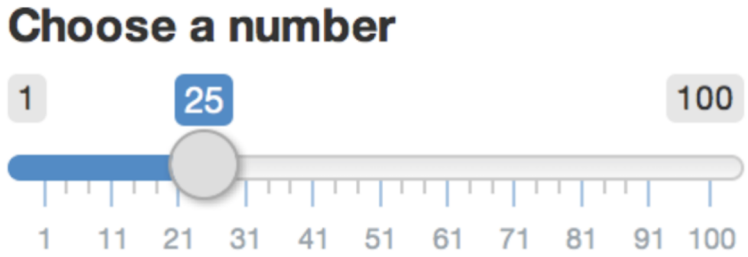

```
library(shiny)

ui <- fluidPage(
…Input()
…Output()
)
server <- function(input, output) {}

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

INPUTS



https://shiny.rstudio.com/gallery/widget-gallery.html

SINTAXIS BÁSICA DE INPUTS




```
ui <- fluidPage(
sliderInput(inputId = "num", label = "Choose a number", …))
```

Tipo de input a utilizar

Input ID (sólo para uso interno)

Etiqueta para mostrar

Otras especificaciones de input

OUTPUTS

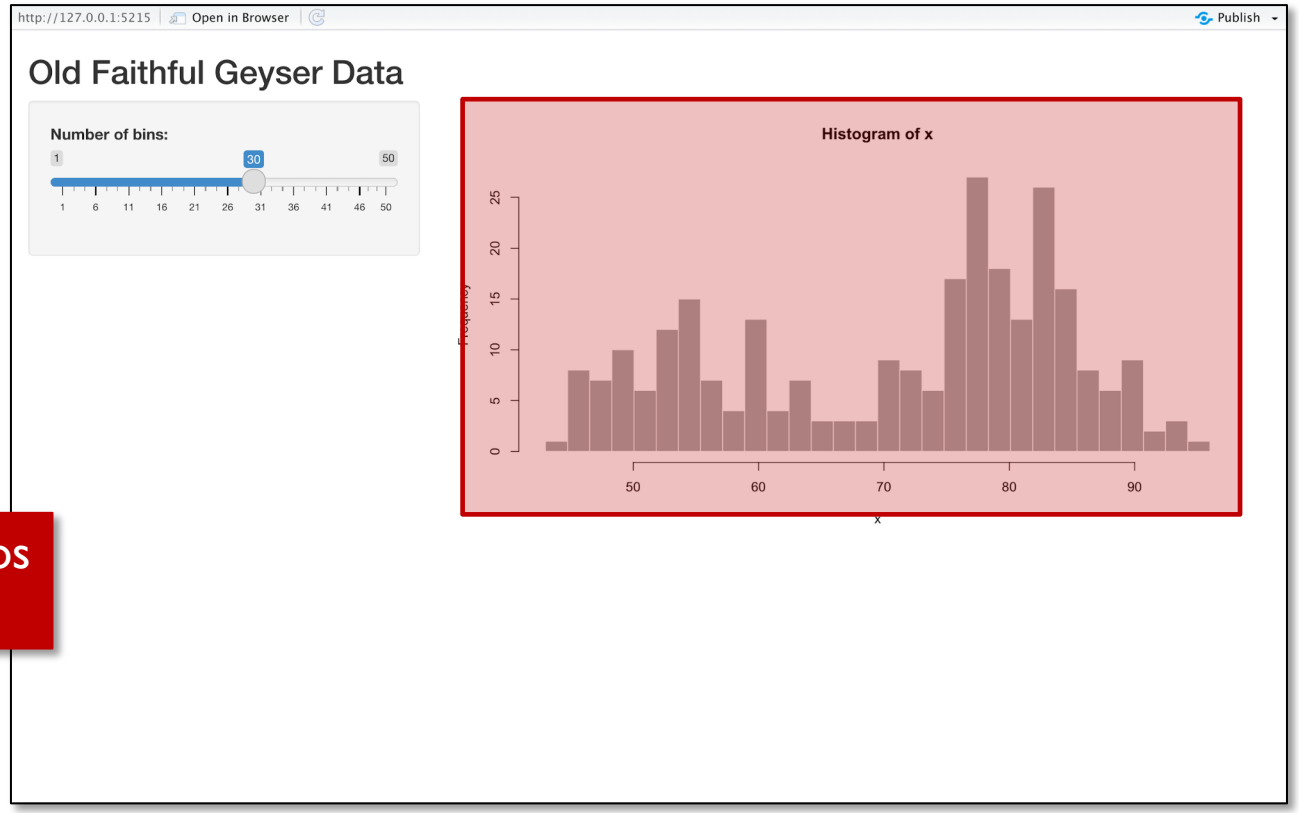
Función	Objeto
dataTableOutput ( )	tabla interactiva
htmlOutput ( )	página web
imageOutput ( )	imagen
plotOutput ( )	gráfica
tableOutput ( )	tabla
textOutput ( )	texto
uiOutput ( )	Elemento de Shiny UI
verbatimTextOutput ( )	Texto
…Output ( )	Otras elementos de paqueterías específicas

SINTAXIS BÁSICA DE OUTPUTS

```
ui <- fluidPage(
plotOutput("hist"))
```

Tipo de output para mostrar

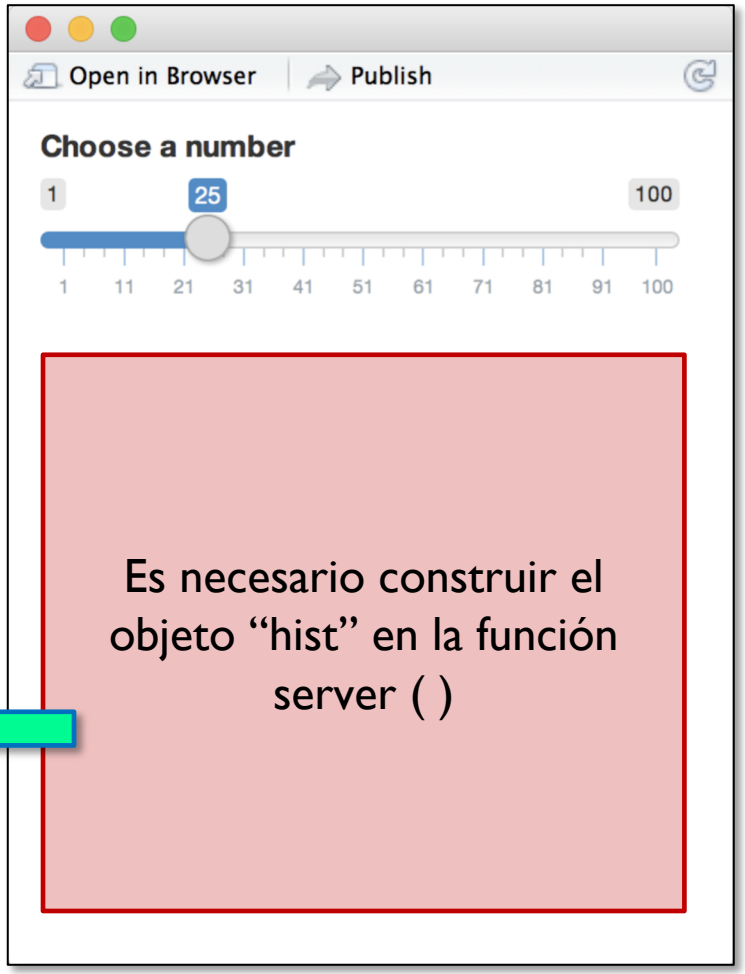
Nombre que le hemos dado al objeto



SINTAXIS DE INPUTS EN LA 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)
```

Agregar coma entre argumentos



Es necesario construir el objeto "hist" en la función server ( )

SERVER : CONSTRUIR LOS INPUTS EN OUTPUTS

```
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)
```

Paso 1. Guardar el objeto para mostrar en **output$**

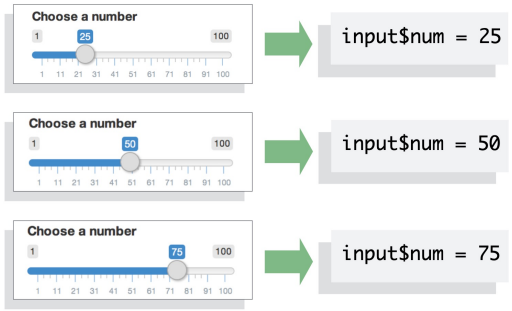
Paso 2. Construir el objeto con **render ( )**

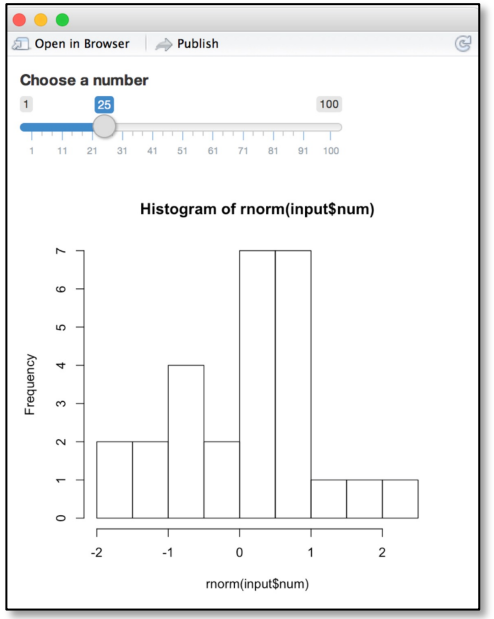
Función	Objeto
renderDataTable ( )	Una tabla interactiva
renderImage ( )	imágen
renderPlot ( )	Un gráfico
renderPrint ( )	Un bloque de código de salida
renderTable ( )	Una tabla
renderText ( )	Una cadena de texto
renderUI ( )	Un elemento de Shiny UI

SERVER : CONSTRUIR LOS INPUTS EN OUTPUTS

```
library(shiny)

ui <- fluidPage(
sliderInput(inputId = "num",
label = "Choose a number",
value = 25, min = 1, max = 100),
plotOutput("hist")
)
server <- function(input, output) {
        output$hist <-  renderPlot ({
        hist(rnom(input$num))
        })
}

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

Paso 3. Ingresar valores nuevos con **input$**



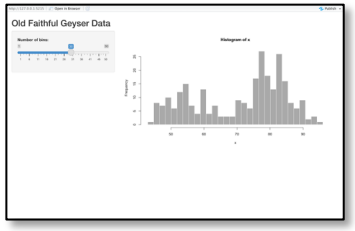


RECAPITULEMOS: INPUTS, OUTPUTS Y SERVER

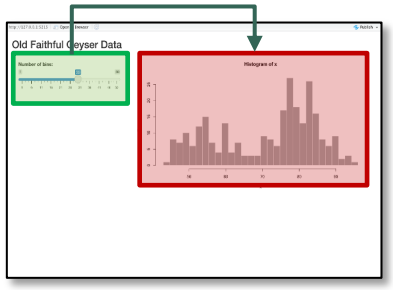
Inputs y Outputs

```
library(shiny)
ui <- fluidPage( )
server <- function(input, output) {}
shinyApp(ui = ui, server = server)
```

Paso 1. Comienza cada app con la plantilla



Paso 2. Utiliza el argumento **fluidpage ( )** para agregar elementos



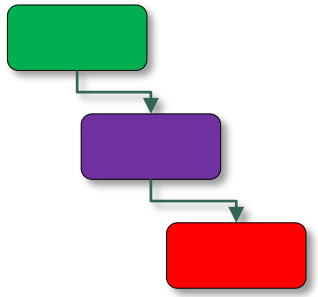
Paso 3. Utiliza las funciones **Input( )** y **Output ( )** para crear elementos reactivos y ensámblalos en la función server

Server

output$hist <- **Paso 4.** Guardamos los objetos que construimos como **output$**

renderPlot ({ }) **Paso 5.** Construimos nuestro objeto con **render ( )**

input$num **Paso 6.** Ingresamos nuevos valores con **input$**



Paso 7. Generar la reactividad usando diferentes **inputs** para construir **outputs renderizados**