Shiny

What is Shiny?

- package to R
- web framework for creating front-end solution for your R code
- easy to build without knowledge of web development

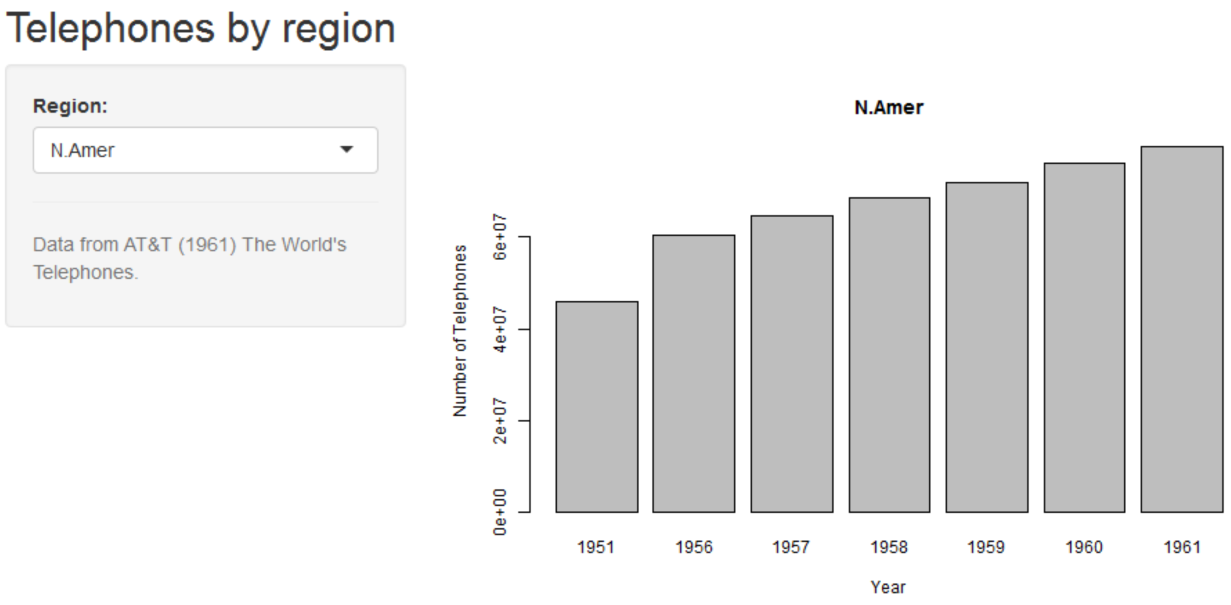
How to install Shiny?

- run in your R console >install.packages(‘shiny’)
- loading package >library(shiny)

Example

Data :

	N.Amer	Europe	Asia	S.Amer	Oceania	Africa	Mid.Amer
1951	45939	21574	2876	1815	1646	89	555
1956	60423	29990	4708	2568	2366	1411	733
1957	64721	32510	5230	2695	2526	1546	773
1958	68484	35218	6662	2845	2691	1663	836
1959	71799	37598	6856	3000	2868	1769	911
1960	76036	40341	8220	3145	3054	1905	1008
1961	79831	43173	9053	3338	3224	2005	1076

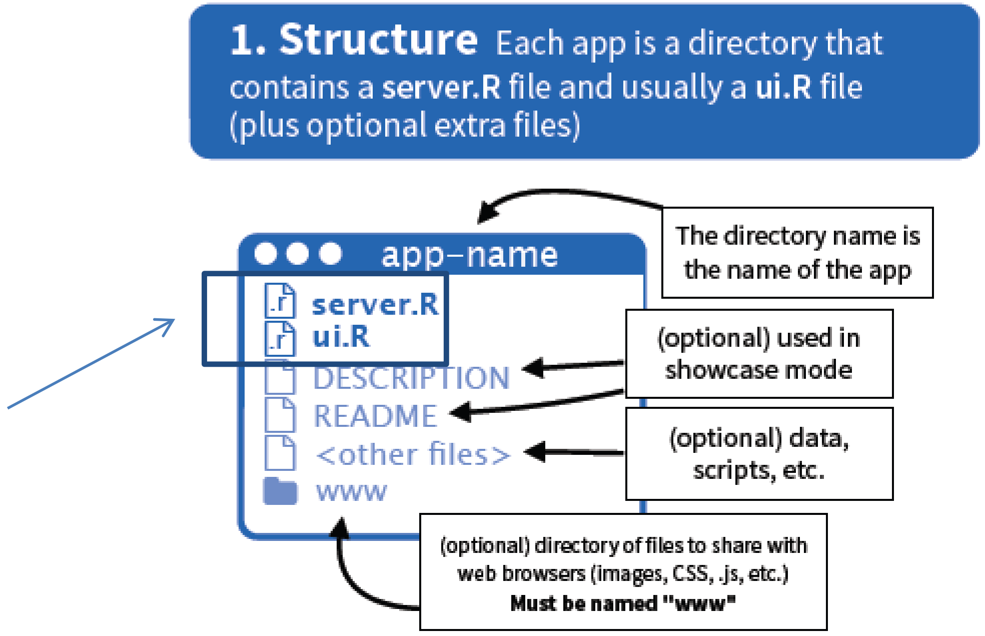

Telephones by region

Region:

N.Amer

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

Structure


1. Structure Each app is a directory that contains a **server.R** file and usually a **ui.R** file (plus optional extra files)

app-name
server.R
ui.R
DESCRIPTION
README
<other files>
www
The directory name is the name of the app
(optional) used in showcase mode
(optional) data, scripts, etc.
(optional) directory of files to share with web browsers (images, CSS, .js, etc.) **Must be named "www"**

server.R or where everything is calculated

Pre-start setup code

```
#server.R
library(shiny)

library(datasets)
```

```
shinyServer(function(input, output) {

output$phonePlot <- renderPlot({

   # Render a barplot
   barplot(WorldPhones[,input$region]*1000,
          main=input$region,
          ylab="Number of Telephones",
          xlab="Year")
  })
})
```

Main shiny function, where every interactive piece should be

server.R or where everything is calculated

Pre-start setup code

```
#server.R
library(shiny)

library(datasets)
```

```
shinyServer(function(input, output) {

output$phonePlot <- renderPlot({

    # Render a barplot
    barplot(WorldPhones[,input$region]*1000,
            main=input$region,
            ylab="Number of Telephones",
            xlab="Year")
  })
})
```

Main shiny function, where every interactive piece should be

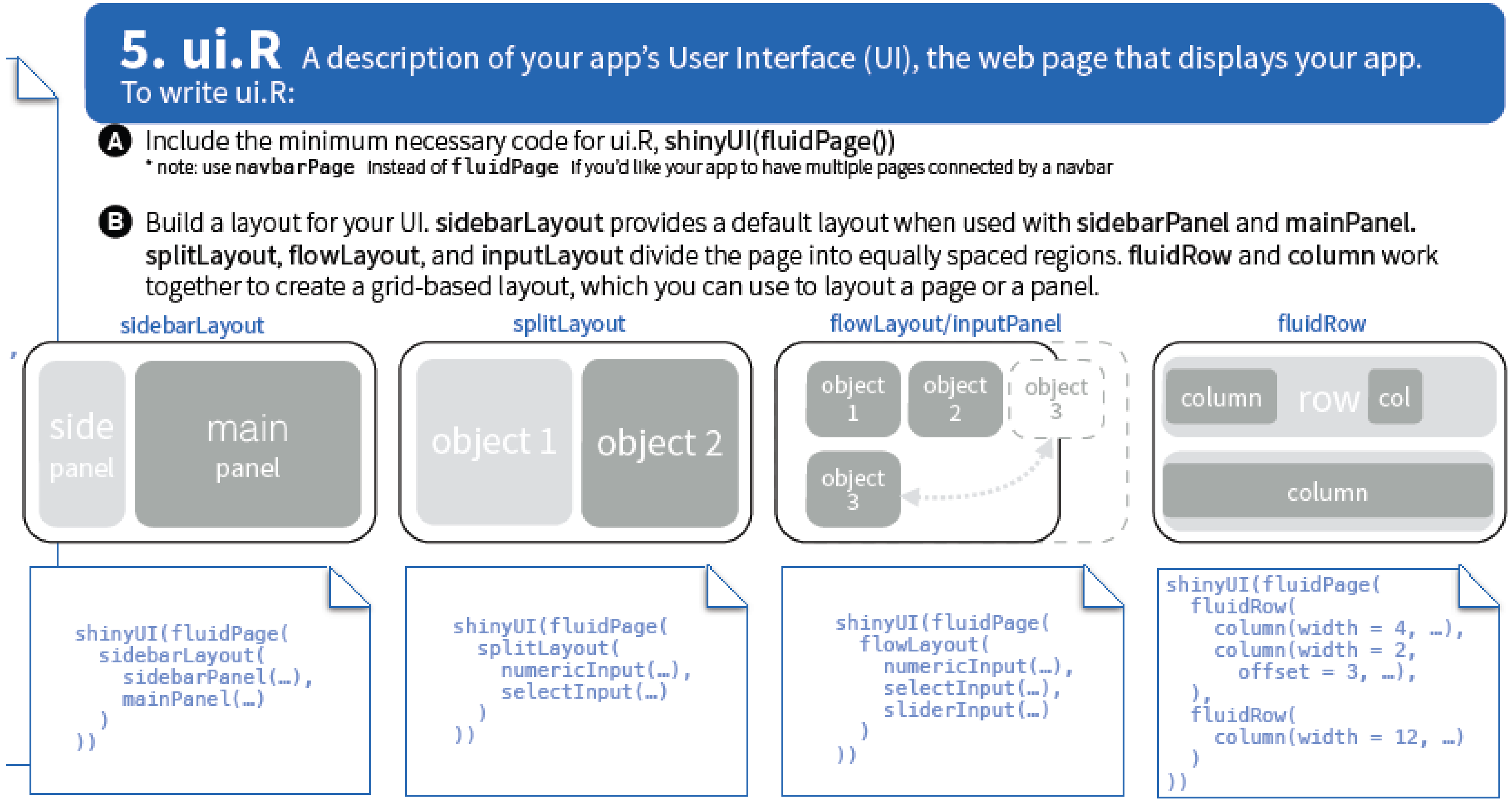
Reactive element - Output

(every time input is changed new plot will be generated)

Reactive element - Input

(every time I choose different region of World Phones input variable will change for that value)

ui.R or how should app looks like



5. ui.R A description of your app's User Interface (UI), the web page that displays your app.
To write ui.R:

A Include the minimum necessary code for ui.R, **shinyUI(fluidPage())**
* note: use `navbarPage` instead of `fluidPage` if you'd like your app to have multiple pages connected by a navbar

B Build a layout for your UI. **sidebarLayout** provides a default layout when used with **sidebarPanel** and **mainPanel.** **splitLayout, flowLayout,** and **inputLayout** divide the page into equally spaced regions. **fluidRow** and **column** work together to create a grid-based layout, which you can use to layout a page or a panel.

```
shinyUI(fluidPage(
  sidebarLayout(
    sidebarPanel(…),
    mainPanel(…)
  )
))
```

```
shinyUI(fluidPage(
  splitLayout(
    numericInput(…),
    selectInput(…)
  )
))
```

```
shinyUI(fluidPage(
  flowLayout(
    numericInput(…),
    selectInput(…),
    sliderInput(…)
  )
))
```

```
shinyUI(fluidPage(
  fluidRow(
    column(width = 4, …),
    column(width = 2,
      offset = 3, …),
  ),
  fluidRow(
    column(width = 12, …)
  )
))
```

ui.R or how should app looks like

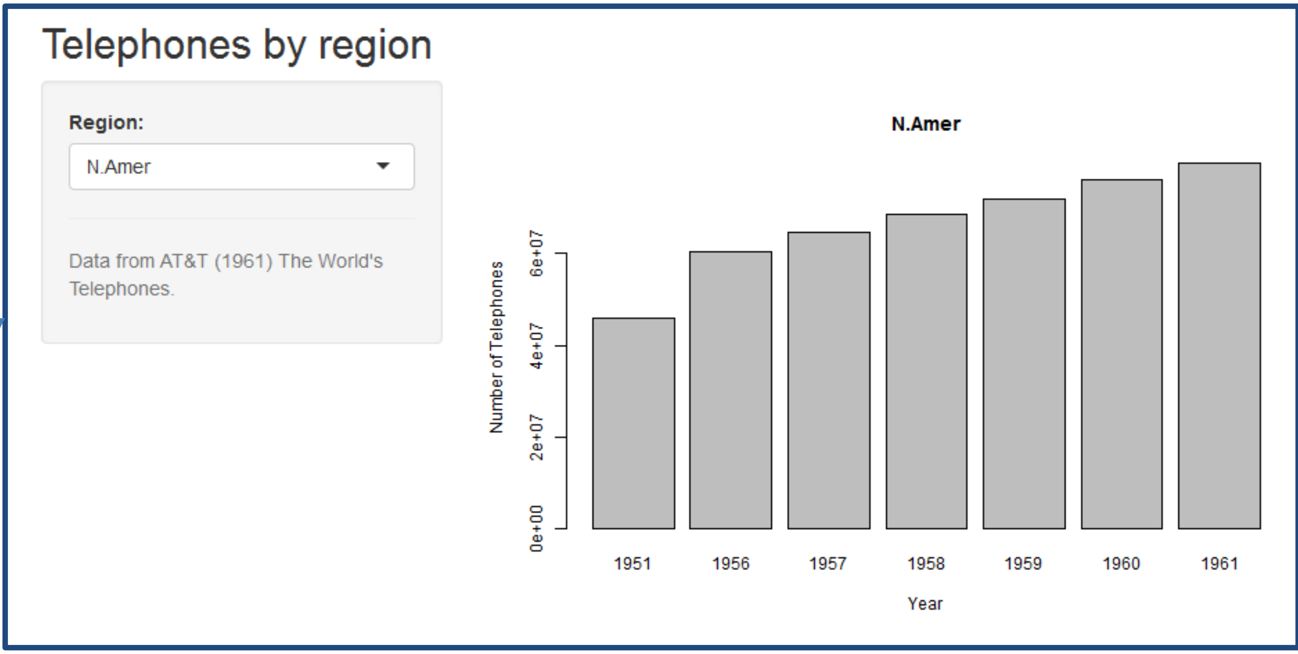
```
#ui.R
library(shiny)
library(datasets)

# Define the overall UI
shinyUI(

 # Use a fluid Bootstrap layout
 fluidPage(

  # Give the page a title
  titlePanel("Telephones by region"),

  # Generate a row with a sidebar
  sidebarLayout(

   # Define the sidebar with one input
   sidebarPanel(
    selectInput("region", "Region:",
           choices=colnames(WorldPhones)),
    hr(),
    helpText("Data from AT&T (1961) The World's Telephones.")
   ),

   # Create a spot for the barplot
   mainPanel(
    plotOutput("phonePlot")
   )

  )
 )
)
```

Main shiny ui function

ui.R or how should app looks like

```
#ui.R
library(shiny)
library(datasets)

# Define the overall UI
shinyUI(

 # Use a fluid Bootstrap layout
 fluidPage(

  # Give the page a title
  titlePanel("Telephones by region"),

  # Generate a row with a sidebar
  sidebarLayout(

   # Define the sidebar with one input
   sidebarPanel(
    selectInput("region", "Region:",
           choices=colnames(WorldPhones)),
    hr(),
    helpText("Data from AT&T (1961) The World's Telephones.")
   ),

   # Create a spot for the barplot
   mainPanel(
    plotOutput("phonePlot")
   )

  )
 )
)
```

Main shiny ui function



Telephones by region

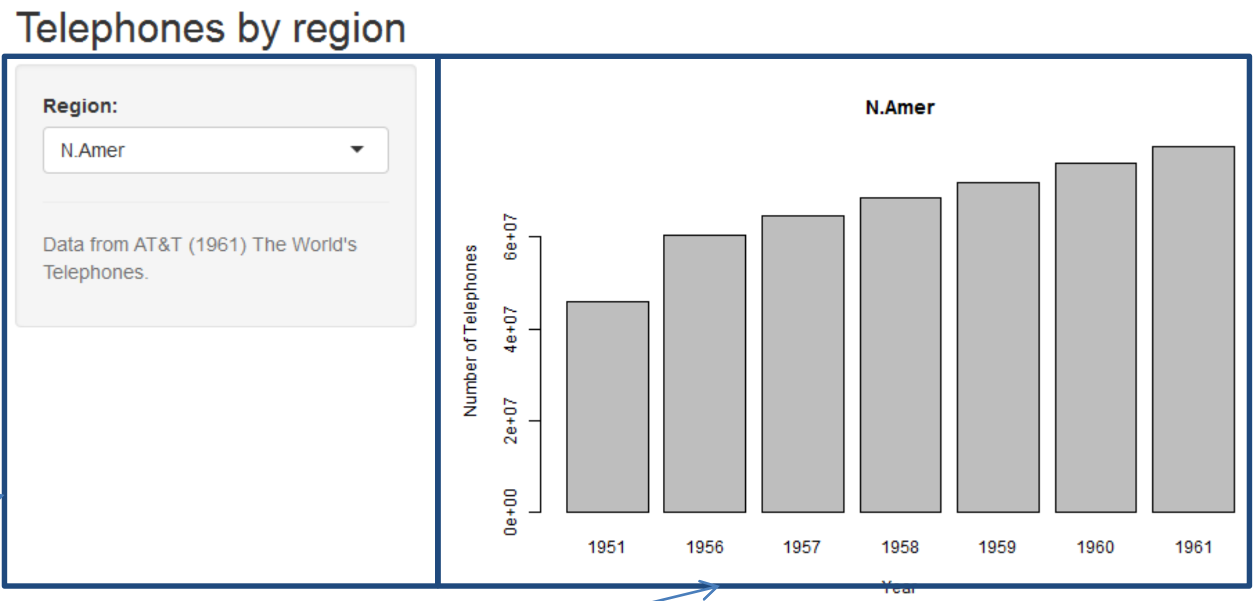
Region:

N.Amer

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

ui.R or how should app looks like

```
#ui.R
library(shiny)
library(datasets)

# Define the overall UI
shinyUI(

 # Use a fluid Bootstrap layout
 fluidPage(

  # Give the page a title
  titlePanel("Telephones by region"),

  # Generate a row with a sidebar
  sidebarLayout(

   # Define the sidebar with one input
   sidebarPanel(
    selectInput("region", "Region:",
            choices=colnames(WorldPhones)),
    hr(),
    helpText("Data from AT&T (1961) The World's Telephones.")
   ),

   # Create a spot for the barplot
   mainPanel(
    plotOutput("phonePlot")
   )

  )
 )
)
```

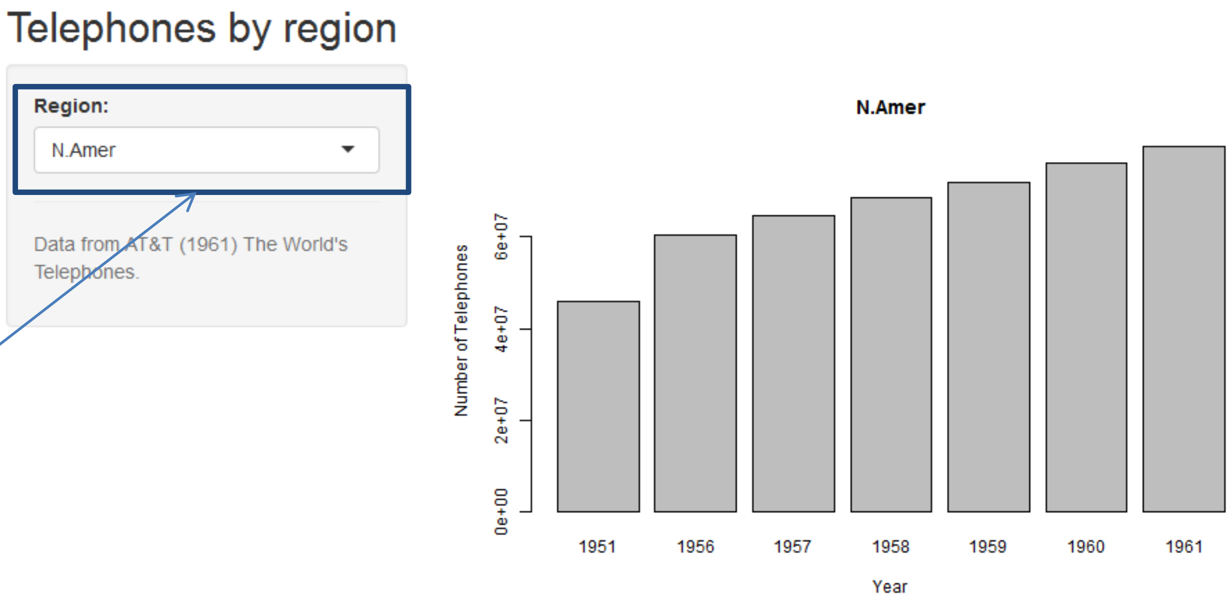

Telephones by region

Region: N.Amer

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

ui.R or how should app looks like

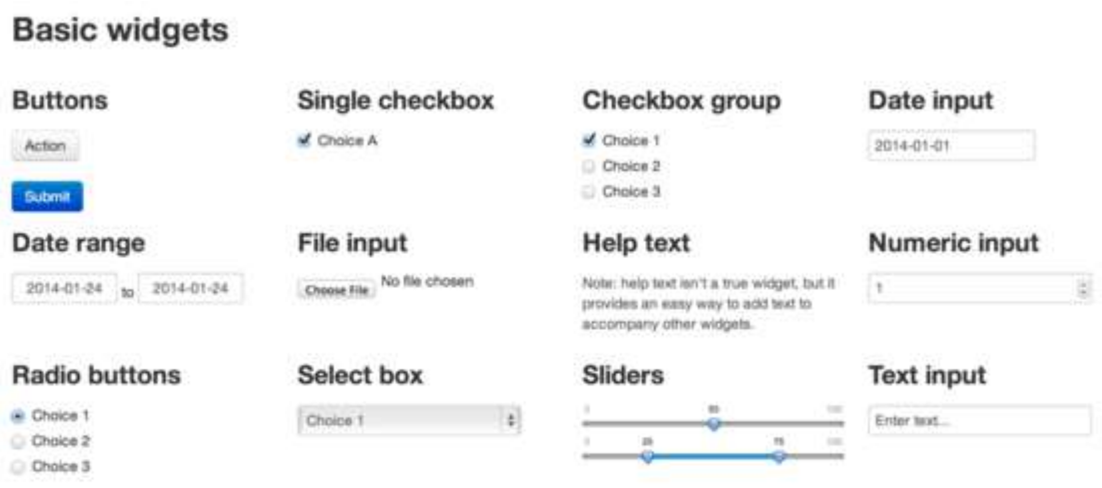
```
#ui.R
library(shiny)
library(datasets)

# Define the overall UI
shinyUI(

 # Use a fluid Bootstrap layout
 fluidPage(

  # Give the page a title
  titlePanel("Telephones by region"),

  # Generate a row with a sidebar
  sidebarLayout(

   # Define the sidebar with one input
   sidebarPanel(
    selectInput("region", "Region:",
          choices=colnames(WorldPhones)),
    hr(),
    helpText("Data from AT&T (1961) The World's Telephones.")
   ),

   # Create a spot for the barplot
   mainPanel(
    plotOutput("phonePlot")
   )

  )
 )
)
```



ui.R or how should app looks like

```
#ui.R
library(shiny)
library(datasets)

# Define the overall UI
shinyUI(

 # Use a fluid Bootstrap layout
 fluidPage(

  # Give the page a title
  titlePanel("Telephones by region"),

  # Generate a row with a sidebar
  sidebarLayout(

   # Define the sidebar with one input
   sidebarPanel(
    selectInput("region", "Region:",
          choices=colnames(WorldPhones)),
    hr(),
    helpText("Data from AT&T (1961) The World's Telephones.")
   ),

   # Create a spot for the barplot
   mainPanel(
    plotOutput("phonePlot")
   )

  )
 )
)
```

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

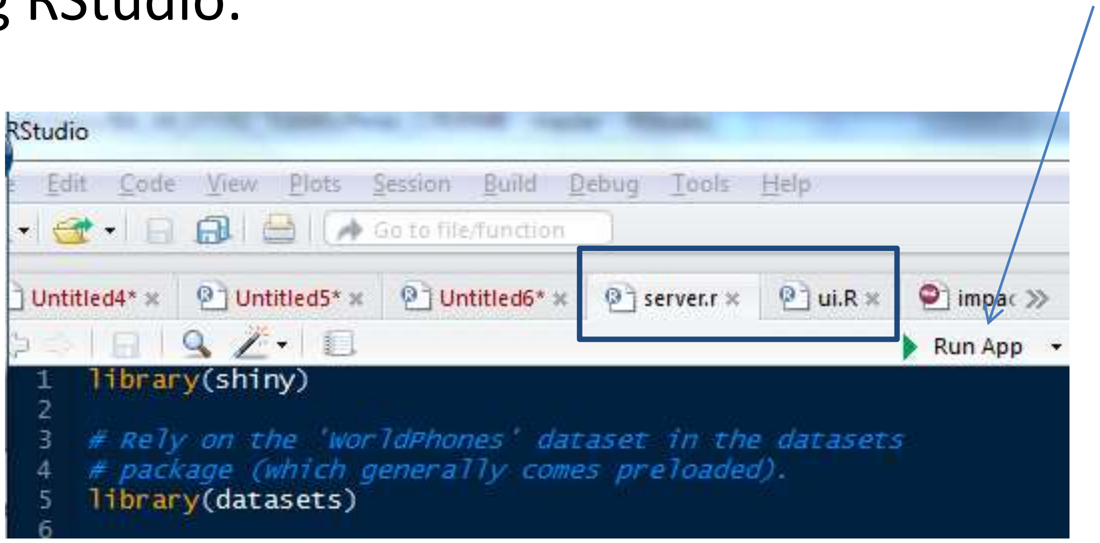


Note: help text isn't a true widget, but it provides an easy way to add text to accompany other widgets.

How to run my Shiny app?

- be sure you are in your shiny app working directory
- type in your R console >shiny::runApp()

- for those using RStudio:

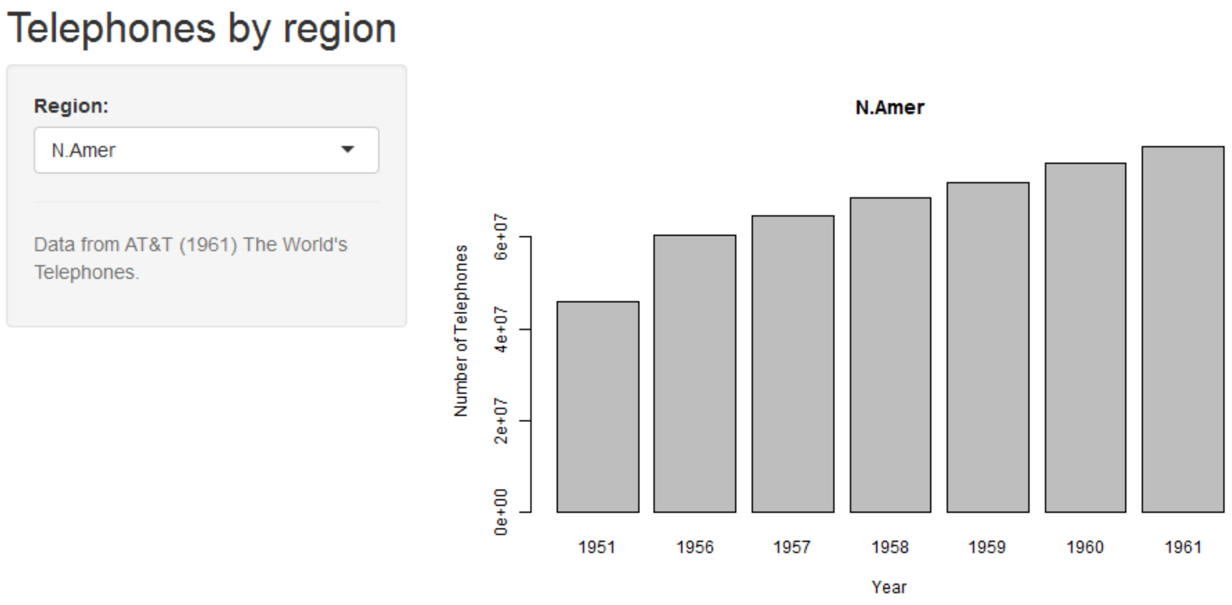


```
library(shiny)

# Rely on the 'worldPhones' dataset in the datasets
# package (which generally comes preloaded).
library(datasets)
```

Example

Data :

	N.Amer	Europe	Asia	S.Amer	Oceania	Africa	Mid.Amer
1951	45939	21574	2876	1815	1646	89	555
1956	60423	29990	4708	2568	2366	1411	733
1957	64721	32510	5230	2695	2526	1546	773
1958	68484	35218	6662	2845	2691	1663	836
1959	71799	37598	6856	3000	2868	1769	911
1960	76036	40341	8220	3145	3054	1905	1008
1961	79831	43173	9053	3338	3224	2005	1076



Telephones by region

Region:

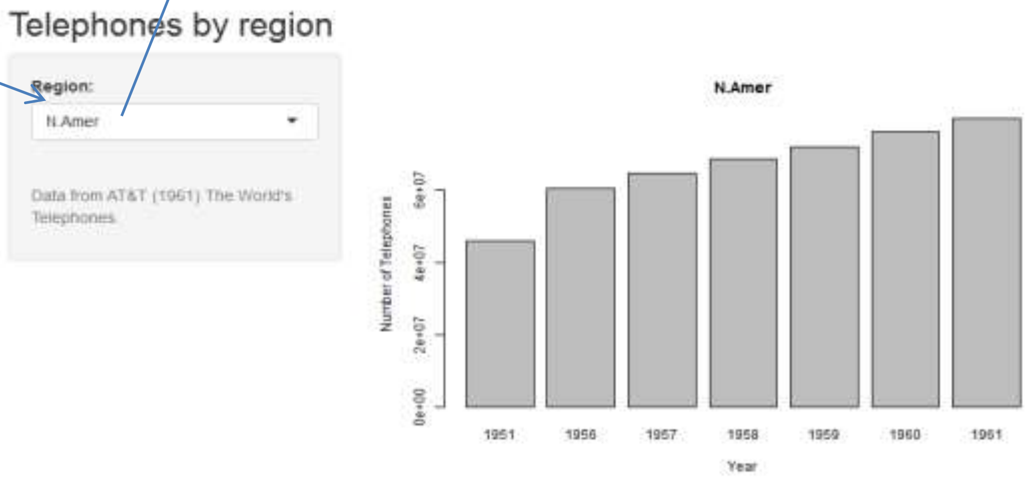
N.Amer

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

Example

```
#ui.R
library(shiny)
library(datasets)

# Define the overall UI
shinyUI(

 # Use a fluid Bootstrap layout
 fluidPage(

  # Give the page a title
  titlePanel("Telephones by region"),

  # Generate a row with a sidebar
  sidebarLayout(

   # Define the sidebar with one input
   sidebarPanel(
    selectInput("region", "Region:",
          choices=colnames(WorldPhones)),
    hr(),
    helpText("Data from AT&T (1961) The World's Telephones.")
   ),

   # Create a spot for the barplot
   mainPanel(
    plotOutput("phonePlot")
   )

  )
 )
)
```

```
#server.R
library(shiny)

# Rely on the 'WorldPhones' dataset in the datasets
# package (which generally comes preloaded).
library(datasets)

# Define a server for the Shiny app
shinyServer(function(input, output) {

 # Fill in the spot we created for a plot
 output$phonePlot <- renderPlot({

  # Render a barplot
  barplot(WorldPhones[,input$region]*1000,
      main=input$region,
      ylab="Number of Telephones",
      xlab="Year")
 })
})
```



Telephones by region

Region:

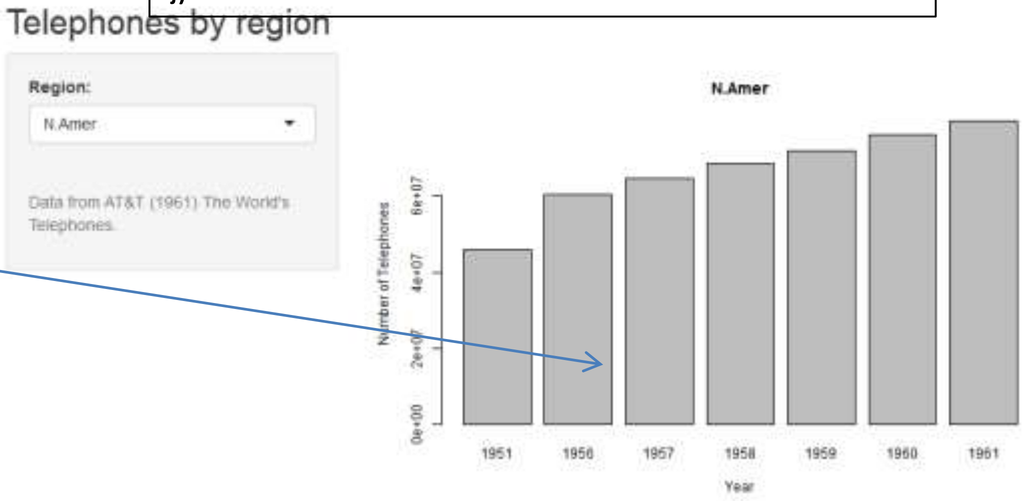
N.Amer

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

Example

```
#ui.R
library(shiny)
library(datasets)

# Define the overall UI
shinyUI(

 # Use a fluid Bootstrap layout
 fluidPage(

  # Give the page a title
  titlePanel("Telephones by region"),

  # Generate a row with a sidebar
  sidebarLayout(

   # Define the sidebar with one input
   sidebarPanel(
    selectInput("region", "Region:",
          choices=colnames(WorldPhones)),
    hr(),
    helpText("Data from AT&T (1961) The World's Telephones.")
   ),

   # Create a spot for the barplot
   mainPanel(
    plotOutput("phonePlot")
   )

  )
 )
)
```

```
#server.R
library(shiny)

# Rely on the 'WorldPhones' dataset in the datasets
# package (which generally comes preloaded).
library(datasets)

# Define a server for the Shiny app
shinyServer(function(input, output) {

 # Fill in the spot we created for a plot
 output$phonePlot <- renderPlot({

  # Render a barplot
  barplot(WorldPhones[,input$region]*1000,
      main=input$region,
      ylab="Number of Telephones",
      xlab="Year")
 })
})
```

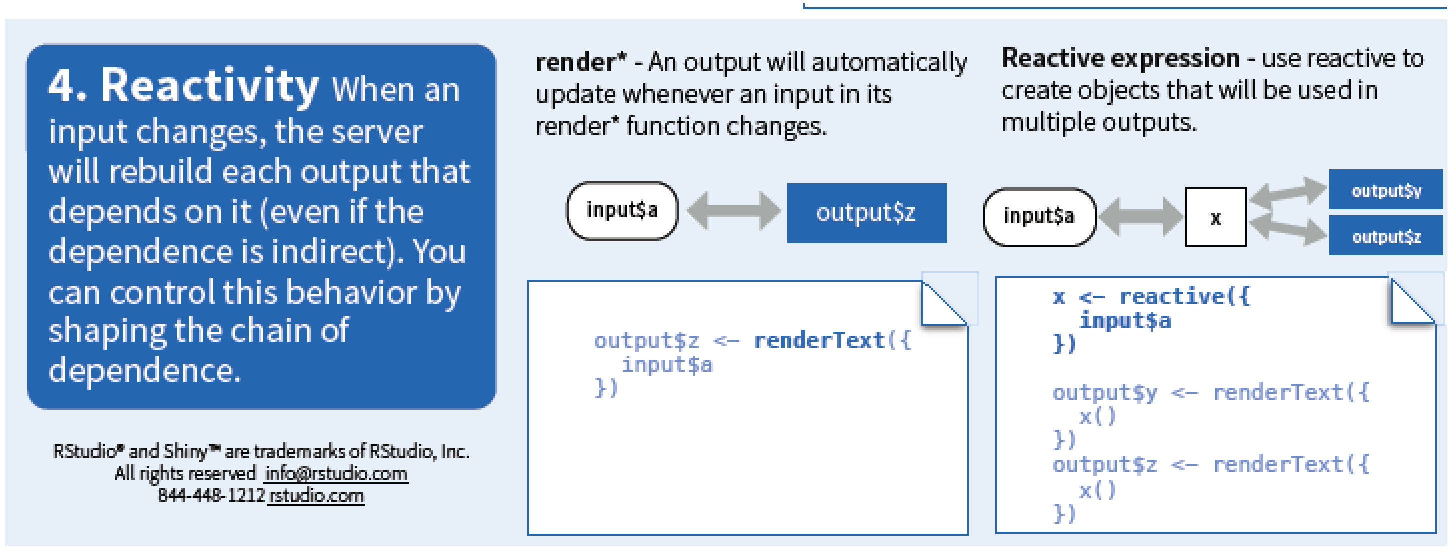

Telephones by region

Region:

N.Amer

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

Reactivity



4. Reactivity When an input changes, the server will rebuild each output that depends on it (even if the dependence is indirect). You can control this behavior by shaping the chain of dependence.

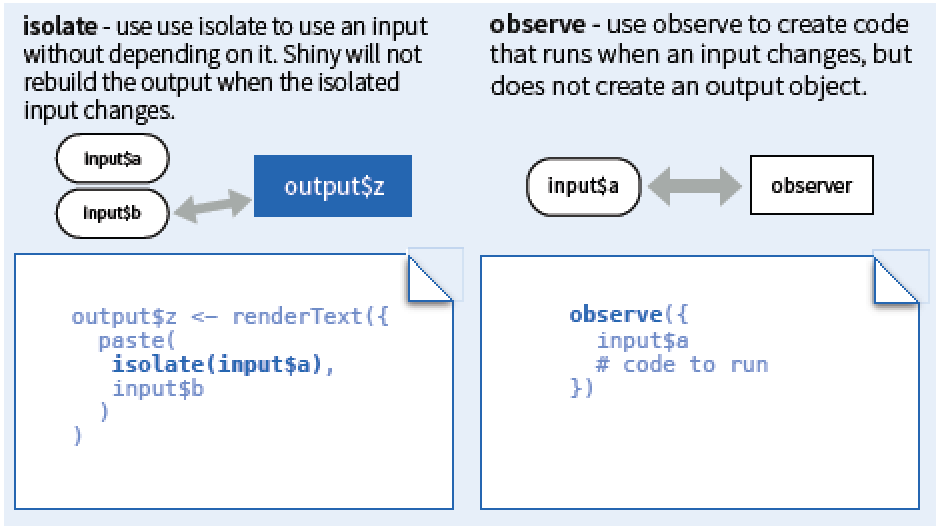
render* - An output will automatically update whenever an input in its render* function changes.

```
output$z <- renderText({
  input$a
})
```

Reactive expression - use reactive to create objects that will be used in multiple outputs.

```
x <- reactive({
  input$a
})

output$y <- renderText({
  x()
})
output$z <- renderText({
  x()
})
```

Reactivity



isolate - use use isolate to use an input without depending on it. Shiny will not rebuild the output when the isolated input changes.

```
output$z <- renderText({
  paste(
   isolate(input$a),
   input$b
  )
)
```

observe - use observe to create code that runs when an input changes, but does not create an output object.

```
observe({
  input$a
  # code to run
})
```

References

- shiny.rstudio.com
- shiny.rstudio.com/gallery/
- shiny.rstudio.com/gallery/widget-gallery.html
- http://shiny.rstudio.com/tutorial/