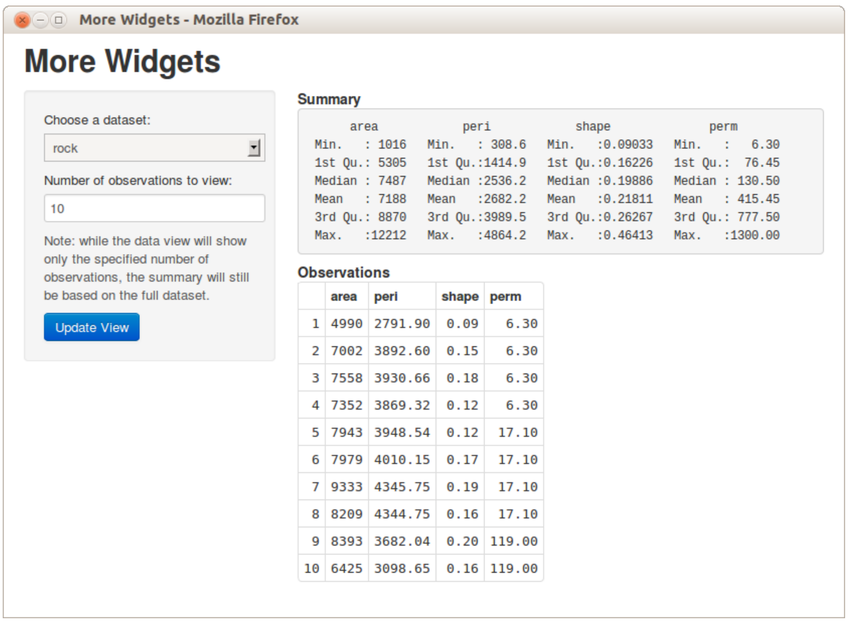
This tutorial is deprecated. Learn more about Shiny at our new location, shiny.rstudio.com.



The More Widgets application demonstrates the help text and submit button widgets as well as the use of embedded HTML elements to customize formatting. To run the example type:

```
> library(shiny)
> runExample("07_widgets")
```

UI Enhancements

In this example we update the Shiny Text application with some additional controls and formatting, specifically:

- We added a `helpText` control to provide additional clarifying text alongside our input controls.
- We added a `submitButton` control to indicate that we don't want a live connection between inputs and outputs, but rather to wait until the user clicks that button to update the output. This is especially useful if computing output is computationally expensive.
- We added `h4` elements (heading level 4) into the output pane. Shiny offers a variety of functions for including HTML elements directly in pages including headings, paragraphics, links, and more.

Here is the updated source code for the user-interface:

ui.R

```
library(shiny)

# Define UI for dataset viewer application
shinyUI(pageWithSidebar(

  # Application title.
  headerPanel("More Widgets"),

  # Sidebar with controls to select a dataset and specify the number
  # of observations to view. The helpText function is also used to
  # include clarifying text. Most notably, the inclusion of a
  # submitButton defers the rendering of output until the user
  # explicitly clicks the button (rather than doing it immediately
  # when inputs change). This is useful if the computations required
  # to render output are inordinately time-consuming.
  sidebarPanel(
    selectInput("dataset", "Choose a dataset:",
                choices = c("rock", "pressure", "cars")),

    numericInput("obs", "Number of observations to view:", 10),

    helpText("Note: while the data view will show only the specified",
             "number of observations, the summary will still be based",
             "on the full dataset."),

    submitButton("Update View")
  ),

  # Show a summary of the dataset and an HTML table with the requested
  # number of observations. Note the use of the h4 function to provide
  # an additional header above each output section.
  mainPanel(
    h4("Summary"),
    verbatimTextOutput("summary"),

    h4("Observations"),
    tableOutput("view")
  )
))
```

Server Script

All of the changes from the original Shiny Text application were to the user-interface, the server script remains the same:

server.R

```
library(shiny)
library(datasets)

# Define server logic required to summarize and view the selected dataset
shinyServer(function(input, output) {

  # Return the requested dataset
  datasetInput <- reactive({
    switch(input$dataset,
           "rock" = rock,
           "pressure" = pressure,
           "cars" = cars)
  })

  # Generate a summary of the dataset
  output$summary <- renderPrint({
    dataset <- datasetInput()
    summary(dataset)
  })

  # Show the first "n" observations
  output$view <- renderTable({
    head(datasetInput(), n = input$obs)
  })
})
```