Crash Course on Shiny Apps

E. F. Haghish
University of Freiburg

8 Feb 2016

What is Shiny?

Shiny is an open source R package that provides an elegant and powerful web framework for building web applications using R. Shiny helps you turn your analyses into interactive web applications without requiring HTML, CSS, or JavaScript knowledge.

What is Shiny?

- Embedding R code in web
- Easy web applications with R
- Creating web-based R products (for point & Click)
- Interactive teaching materials
- Interactive presentations

Who should really bother with Shiny?

Unless you want to build a web application to process some data using R – where doing the same computation is rather more costly to do with another language (which consequently, requires updating and presenting the results) – **you don't really need Shiny**. But there are more benefits in Shiny for Statistics Teachers.

- Interactive teaching materials with web-based GUI interface
- Interactive presentations (*Minor and Fancy*)

What is Shiny?

- It includes several pre-made web programming functions
- The functions are also used for designing the web application
- It doesn't require you to know **HTML**, **CSS** (Cascading Style Sheets), or **JavaScript**. But knowing some web programming is a bonus.

What is Shiny?

- But I also think **Shiny is not as shiny as the say**
- It has certain drawbacks.
  - It's rather complicated
  - The User Interface is built using functions
  - You will struggle a bit more if you're not a web programmer

About the current lecture

- In this lecture I introduce Shiny by focusing on **creating interactive presentation** and **teaching statistics**. Developing web applications follows the same procedure that I discuss in this lecture, but I barely think you'd be interested in it.
- Using Shiny for teaching and creating interactive figure presentations is only reasonable if the **computation time is instant**. For example, if you want to create:

Installation

Shiny is a package, which can be installed and laoded as follows.

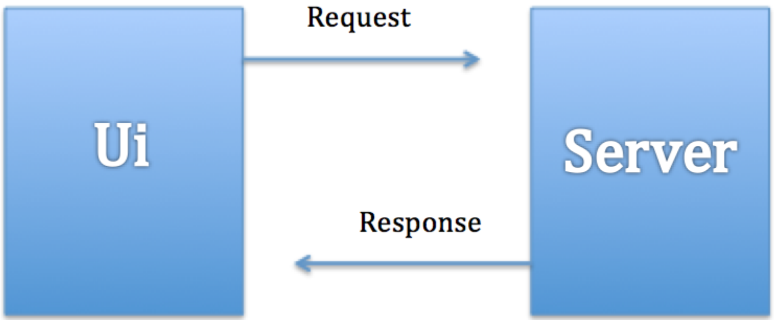
```r
install.packages("shiny")
library(shiny)
```

Shiny App Structure

- Uses R as a server
- R would be listening to the web application
- If you use your laptop as server (run it from your computer) you won't be able to type any other command while the app is running

Shiny App Structure

The main app structure includes a **User Interface** and a **Server**

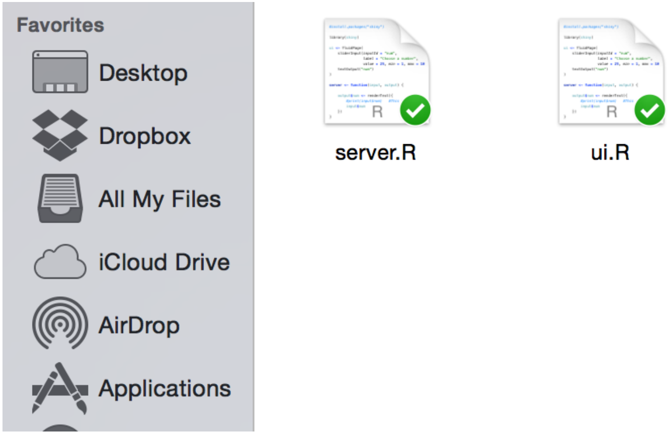


The Old Structure

Shiny has two way for creating an application. The older method was more complicated (weird) because it required making 2 separate R scripts named, each included 1 main function

- **ui.R** creates the User Interface with **shinyUI** function
- **server.R** creates the server with **shinyServer** function

The Old Structure



The New Structure

The new syntax allows creating Shiny within a single R script file. The apps with the old structure can be transformed to the new syntax as follows. *This is rather important because the new syntax is more convenient but most of the material online are written witht the old syntax*

- define a **ui** object instead of `shinyUI()`
- define a **server** object instead of `shinyServer()`
- connect the objects using **shinyApp()** function

Shiny App Template (Example_00)

```r
library(shiny)

#Part 1: User Interface Object
ui <- fluidPage{

    }

#Part 2: Server
server <- function(input, output) {

    }

#Part 3: Connecting the Ui and the Server
shinyApp(ui = ui, server = server)
```

Example 01 (Hello IMBI!)

The **titlePanel()** function prints a Heading 1. So we can write the classic example of Hello World! In this case, hello IMBI!

Example 02

There are other functions that can be used for writing different HTML tags. Each function is usually named after a tag. For example **
** tag is named **h1()**. and so on...

Can we make it easier? (Example 03)

Using multiple functions for writing text is more complex than using the actual HTML tags. But things can be **much easier** if we write a **Markdown** file, similar to RMarkdown, and import it in Shiny. Example_03 demonstrates how to use the `includeMarkdown()` function to load a Markdown file in the User Interface of the App.

The First App (Example 04)

The **example_04** is the first actual app. But yet, very simple. it merely includes a slide bar, which ranges from 1 to 100, and after moving the slide, it **squares** the value and prints the prints it. This might sound simple, but requires a complete setup between the User interface (for slide, and printing the output) and the server, for squaring the value and passing it to the user interface.

Second App (Example 05)

Let's use one of the R's data sets to create an app that works with Data. For this, we can use some of the **Example Datasets in R** to play with. for example the **car** dataset. Which has 2 variables, **speed** and **dist**.
If you type `head cars`

```
> head(cars)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
```

Second App (Example 05)

In this example I will create a slide bar, similar to the previous examples. However, I will create **2 outputs** for the given number. Using the `cars` example dataset:

- First I create histogram of the `speed` variable (the first column) of the cars data set. I will use the number from the slide bar to change the number of `bins` of the histogram (i.e. number of bars in the plot)
- I use the given number again to draw a random sample from the **cars** data set without replacement. And show it in a scatter plot.
- I also make another change in this example. I create a **side bar** and **main panel** which divides the window in 2 parts. The **side bar** will include the slide, and the **main panel** will show the R output.

Get deeper in Shiny

- More examples `http://shiny.rstudio.com/gallery/`
- Most of the examples in the gallery, are focused on “the app” instead of thinking about other applications such as **Education**. In my examples, I tried to cover the part about adding text in the app conveniently. The same procedure for inserting text in the app can be applied to modify any of the complex examples in the Shiny App gallery.
- If you need help, you know where to reach me!