

Group Meeting: Shiny Analyses → Interactive Apps

Sahil Shah
sahil.shah@u.northwestern.edu

June 8, 2016

Meeting materials available on: github.com/sahildshah1/shiny-groupmtg

Introduction

Install/Run Shiny apps

Shiny is an R package that makes it easy to build apps

Install Shiny

- `install.packages("shiny")`

Running a Shiny app

- If Shiny app is in a directory called `my-app`:
 1. `library(shiny)`
 2. `runApp('my-app')`
- R session will be busy while Shiny app is active
- Run another Shiny app → Open another R session
- Get R session back → Hit `esc`

Examples of Shiny apps

Examples built into Shiny package: Run from R or view online

- [http://shiny.rstudio.com/tutorial/lesson1/#Go Further](http://shiny.rstudio.com/tutorial/lesson1/#Go_Further)
- Each demonstrates a feature of Shiny apps
- Examples open in 'showcase' mode with the ui.R and server.R scripts in the display.

Gallery of examples available online:

- <http://shiny.rstudio.com/gallery/>
- Contains useful examples to learn from

`shiny.rstudio.com/*`

- Tutorial → `shiny.rstudio.com/tutorial`
- Articles → `shiny.rstudio.com/articles`
- NB 'cf. *.html' in the rest of the slides are `shiny.rstudio.com` pages

Google + Stack Overflow

Build a Shiny app

'Hello Shiny' Example

View example online

<http://shiny.rstudio.com/tutorial/lesson1/>

OR

Run example in R

```
library(shiny)  
runExample("01_hello")
```

Structure of a Shiny app

Two R scripts saved together in a directory

- Each app needs its own directory
- `ui.R` : controls layout and appearance of app
- `server.R`: contains instructions needed to build app

'Hello Shiny' example layout

- `fluidPage`, `sidebarLayout` → Define layout
- Customize layout → cf. [articles/layout-guide.html](#)

Place widgets and R objects in layout

- widgets (eg slider) → cf. tutorial/lesson3
 - Input functions create widgets
 - Access widget value in R object with the `label` arg
- R object (eg plot) → cf. tutorial/lesson4
 - Output functions create R objects
 - The arg `Output` has to match label in `server.R`

Build R objects in shinyServer function

- The output 'list' stores R objects
 - render function: instructions to create object → tutorial/lesson4
 - Element name labels R object and should match argument of Output function in `ui.R`
- The input 'list' stores values of widgets
 - Call a widget value inside a render function using the label of the widget (Input function argument)

Where you put code in server.R determines how many times run

Execution of commands in server → cf. tutorial/lesson5/

- **Outside of shinyServer:** Run once (eg. library, load, source)
- **In shinyServer:** Run once a user visits app (eg. record session info)
- **In render:** Run each time user changes a widget

Customize/ Manage larger apps

App with multiple distinct sub-components → Navbar Pages and Fluid Grid

- <http://shiny.rstudio.com/articles/layout-guide.html>
- <http://shiny.rstudio.com/gallery/navbar-example.html>
- navbarMenu cannot be the first item in the navbarPage

Search-able/Sort-able table from dataframe → DataTable

- <http://shiny.rstudio.com/gallery/basic-datatable.html>
- <https://rstudio.github.io/DT/shiny.html>

Modularizing Shiny

- <http://shiny.rstudio.com/articles/modules.html>
- `source *.R files → source(.,local=TRUE)`
<https://stat.ethz.ch/R-manual/R-devel/library/base/html/source.html>
- `source*.R files → source()$value`
<http://stackoverflow.com/questions/30534674>
- cf. `skeleton-app`