Shiny: Part 1

Daniel Anderson
Week 8, Class 1

Agenda

- Shiny!
  - template
  - ui/server

Learning objectives

- Be able to create basic interactive plots with shiny
- Understand basic layouts

It's required you make this joke when talking about shiny

(especially if you're a parent of a 5 and 7 year old)

The basics!

Do the following:

- Create a new R project
- Select "Shiny Web Application"
- Select the directory
- Click "create project"

(demo)

First thing - Run App!

- I like to open in the browser
- What's going on here? Talk with your neighbor and see if you can deduce the basic functionality.

02:00

UI

- The `ui` defines the look and feel of the app - the user interface
- Use it to define where output "lives"
- It also defines the inputs for the `server`, where functions are actually evaluated.
- In this case - we have defined a `sliderInput`, which we're calling `"bins"`. It will take on values from 1 to 50, and will start at 30.
- Access the specific value the user selects within the `server`, through `input$bins`.

Server

- The `server` function takes the input from the UI and puts it in normal R code.
- In this case, we're creating one output object, called `distPlot`. The result is then called through the `ui` on line 30
- Try removing lines 39-44, and replacing it with `print(input$bins)`
- Feels like it should work, but it doesn't. Any ideas why?
- We are no longer rendering a plot. We are rendering text. We need to change `renderPlot` to `renderText` and `plotOutput` to `textOutput`.

Does it work now?

Challenge

- Try producing essentially the same thing as the default, but use `ggplot` instead of base plotting.
- Reminder, the data are called `faithful`, and you'll be showing the distribution of `waiting`

10:00

[demo]

Change the input

- Let's say instead of a slider, we want 5 options: 1, 5, 20, 50, or 100
- We can change the input from a slider (representing continuous intervals) to radio buttons, representing discrete choices.
- Just change `sliderInput` to `radioButtons`, and put in the appropriate `choices`. You can also add a `selected` argument for which it defaults to.
- Tricky - it will treat the input from `radioButtons` as a character/factor, so you have to override this behavior.

Try it out!

05:00

[demo]

More complicated

- Let's say we wanted to build a plot that showed the distribution of highway miles per gallon that could be faceted by `year`, `trans`, or `class`.
- Let's say you also wanted to keep a slider on there to control the number of bins.
- Use `radioButtons` for the variables, and `sliderInput` for the bins.
- You'll need to create a second entry to your `ui`

(demo)

Tables

- My recommendation - use the `DT` package, with `DT::datatable`.
- Sort of silly example - let's build a table to go below our distribution(s) that shows `mean`, `sd`, `min`, and `max` of the corresponding distribution(s)
- You'll need to create a new output object in the server, and tell the ui where it should be rendered
- Use `renderDataTable` in the server and `dataTableOutput` in the ui
- If you want to use `{dplyr}`, as I would, then you have to deal with NSE. In this case it's not that big of a deal though. Just use `!!sym(input$var)`, where `sym` transforms a string to a symbol.

(demo)

Last thing for today

Change the layout

- Let's say instead of having the table below, we wanted it within a tabset
- Just create a `tabsetPanel` within the `mainPanel`, then put the output for each tab within `tabPanel`. (demo)

Alternative

- Instead of using a tabset with `tabsetPanel`, you might want to have a navbar at the top of the page, which you can create with `navbarPage`.
- Can be a bit more complicated - each `tabset` needs to include everything, including the `sidebarPanel` (if present), could include tabsets, `mainPanel`, etc.
- Essentially each tab from the `navbar` becomes an entirely new page.
- Consider pulling pieces out as a refactoring method to try to make code more readable/less buggy.

(example)

Conclusions

- Shiny is super customizable - just scratched the surface today
- Great for building interactive plots, but you can use it for all sorts of other things too (including text and tables)
- Next time we'll talk about {shinydashboard} for, basically, an alternative layout.
- If you end up wanting to go deep with shiny, you may want to read more about reactivity