PA 446

Coding for Civic Data Applications

Will be starting at 6:05pm

Class #12

Logistics

Course Logistics

- Final project rubric
- HW5
  - Due before class next week
  - Progress on final project
- Midterm
  - General feedback and Q&A on 11/17

Final Assignment Rubric

Category	Description	Percent of Grade
Problem solving	You clearly documented your audience, data source, and the problem you are trying to solve with your Shiny app	10
Quality of data wrangling	You clearly documented and wrote code to explore and clean the data, ensuring data quality issues will not significantly impact analysis	15
Quality of analysis	I want to see that you put thought into the type of analysis	15
User experience and graphics	Your Shiny app runs and has no errors. Clear input and output for the user. The Shiny app allows your audience to answer the problem you laid out.	20
Basic Shiny	Your Shiny code is appropriated divided into UI, server, and any other needed chunks Your Shiny code should be clean and easy to read Your Shiny app should include usage of functions	20
Intermediate Shiny	You are using reactives to minimize the number of times you read in the same data You are using ggplot to create advanced data viz You are organizing the app so the user can easily navigate and so reviewers of your code knows which code refers to what	20

Final Assignment Rubric

Reference prior class slides for best practices and ideas

Category	Description	Percent of Grade
Problem solving	You clearly documented your audience, data source, and the problem you are trying to solve with your Shiny app	10
Quality of data wrangling	You clearly documented and wrote code to explore and clean the data, ensuring data quality issues will not significantly impact analysis	15
Quality of analysis	I want to see that you put thought into the type of analysis	15
User experience and graphics	Your Shiny app runs and has no errors. Clear input and output for the user. The Shiny app allows your audience to answer the problem you laid out.	20
Basic Shiny	Your Shiny code is appropriated divided into UI, server, and any other needed chunks Your Shiny code should be clean and easy to read Your Shiny app should include usage of functions	20
Intermediate Shiny	You are using reactives to minimize the number of times you read in the same data You are using ggplot to create advanced data viz You are organizing the app so the user can easily navigate and so reviewers of your code knows which code refers to what	20

HW5

Have gone through at least 1 pass and code + documentation to show for it

Category	Description	Percent of Grade
Problem solving	You clearly documented your audience, data source, and the problem you are trying to solve with your Shiny app	10
Quality of data wrangling	You clearly documented and wrote code to explore and clean the data, ensuring data quality issues will not significantly impact analysis	15
Quality of analysis	I want to see that you put thought into the type of analysis	15
User experience and graphics	Your Shiny app runs and has no errors. Clear input and output for the user. The Shiny app allows your audience to answer the problem you laid out.	20
Basic Shiny	Your Shiny code is appropriated divided into UI, server, and any other needed chunks Your Shiny code should be clean and easy to read Your Shiny app should include usage of functions	20
Intermediate Shiny	You are using reactives to minimize the number of times you read in the same data You are using ggplot to create advanced data viz You are organizing the app so the user can easily navigate and so reviewers of your code knows which code refers to what	20

HW5

Need a write up of your analysis plan

Category	Description	Percent of Grade
Problem solving	You clearly documented your audience, data source, and the problem you are trying to solve with your Shiny app	10
Quality of data wrangling	You clearly documented and wrote code to explore and clean the data, ensuring data quality issues will not significantly impact analysis	15
Quality of analysis	I want to see that you put thought into the type of analysis	15
User experience and graphics	Your Shiny app runs and has no errors. Clear input and output for the user. The Shiny app allows your audience to answer the problem you laid out.	20
Basic Shiny	Your Shiny code is appropriated divided into UI, server, and any other needed chunks Your Shiny code should be clean and easy to read Your Shiny app should include usage of functions	20
Intermediate Shiny	You are using reactives to minimize the number of times you read in the same data You are using ggplot to create advanced data viz You are organizing the app so the user can easily navigate and so reviewers of your code knows which code refers to what	20

HW5

You will be graded on completion for HW5

Set up office hours with me to brainstorm ideas if you are stuck

HW5

- Submit everything in an R file
- Submit on Blackboard

Class #12

Shiny Continue

Clarification from Last Class

```
Listening on http://127.0.0.1:7299
```

The http port access only works for you (other people cannot visit your app)

- To make your app visible to others, you will need to find a hosting service

UI vs Server: General Differences

	UI	Server
Computing resources	Provided by the users of your app	Provided by you (host)
Nomenclature	Front end	Back end

UI

Main components

Manages the application's inputs and outputs

- **Inputs**, from the user
- **Outputs**, from the server function

Input function

There are countless variations

These are just some common ones

[to coding]

Output function

Less variation

There are three main types of output

- Text
- Tables
- Plots
    - Maps (we will not be covering)

Output functions on the front end (UI) has to be coupled with a render function in the back end (server)

Output function

Text

For text, tables and plots, there are usually 2 methods:

- textOutput(): rendering plain text
- verbatimTextOutput(): “fixed code and console output” , aka less popular

[we will not show code for this]

Output function

Table

tableOutput(): static table. Most useful for small, fixed summaries (e.g. model coefficients)

dataTableOutput(): dynamic table. Gives programmer and user more control of what you show/see and most appropriate if you want to expose a complete data frame to the user

[to coding]

Plotting

You can display any type of R graphic (base, ggplot2, etc) with plotOutput + renderPlot (from server)*

*You cannot easily render maps this way, and will need to rely on additional packages

[to coding]

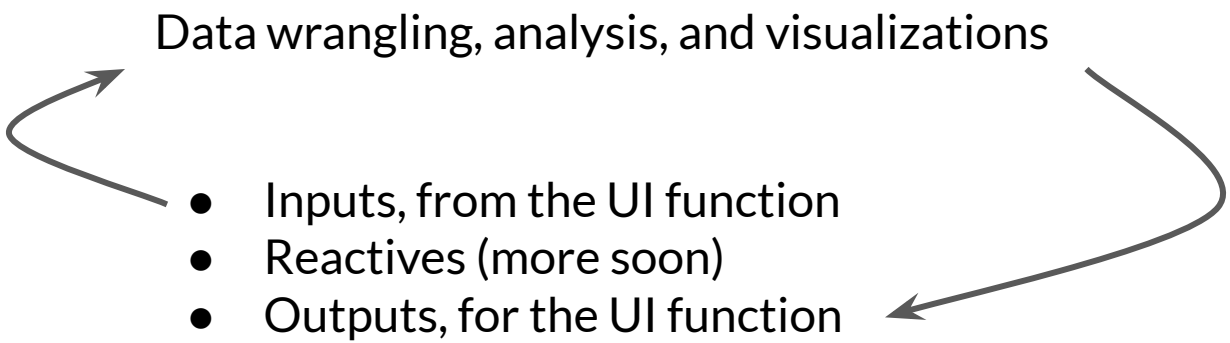
Server

Main components

Data wrangling, analysis, and visualizations

- Inputs, from the UI function
- Reactives (more soon)
- Outputs, for the UI function

Main components



Inputs

For the server, the input is an object, not a function

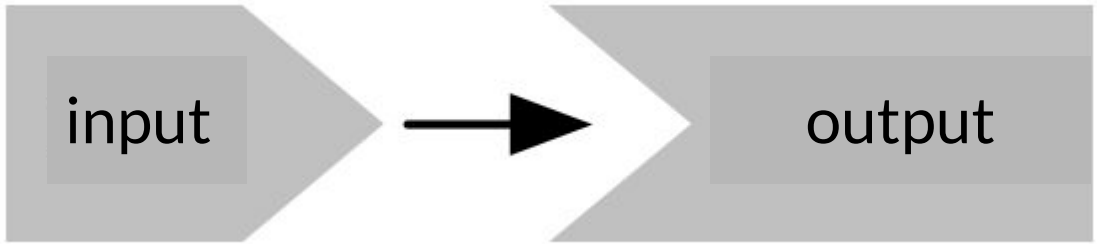
- Input objects are read only
- Inputs can only be read in a render function or reactive()

Outputs

- Similar to inputs, you always use the output object with a render function
  - Note that the {} are only required in render functions if need to run multiple lines of code

- If you misspell an output file, it will not give you a warning

Relationship Between Input and Output



Reactivity

Reactivity Example

The search bar on:
https://www.uic.edu/
https://web.mit.edu/

[to coding]

Imperative v Declarative Programming

Imperative programming: you “issue a specific command and it’s carried out immediately.” What you have done so far in R

Declarative programming: you “express higher-level goals or describe important constraints, and rely on someone else to decide how and/or when to translate that into action.”

Imperative v Declarative Programming

Imperative programming

- Most programming in R
- Order matters

For Declarative programming

- Most programming in Shiny
- Order doesn't matter
    - the order in which reactive code is run is determined only by the reactive graph, not by its layout in the server function.

Reactives

Example:

```
filtered_df <- reactive(input$df_salaries %>%
                          filter(Department==input$department_input))
```

Reactives

- “Like inputs, you can use the results of a reactive expression in an output.
- Like outputs, reactive expressions depend on inputs and automatically know when they need updating.”

In practice, you can treat them like a variable

Reactive Expression

Purpose

- Minimize recomputation when inputs change, making apps more efficient
- Make it easier for humans to understand the app by simplifying the reactive graph
- Standardize source data

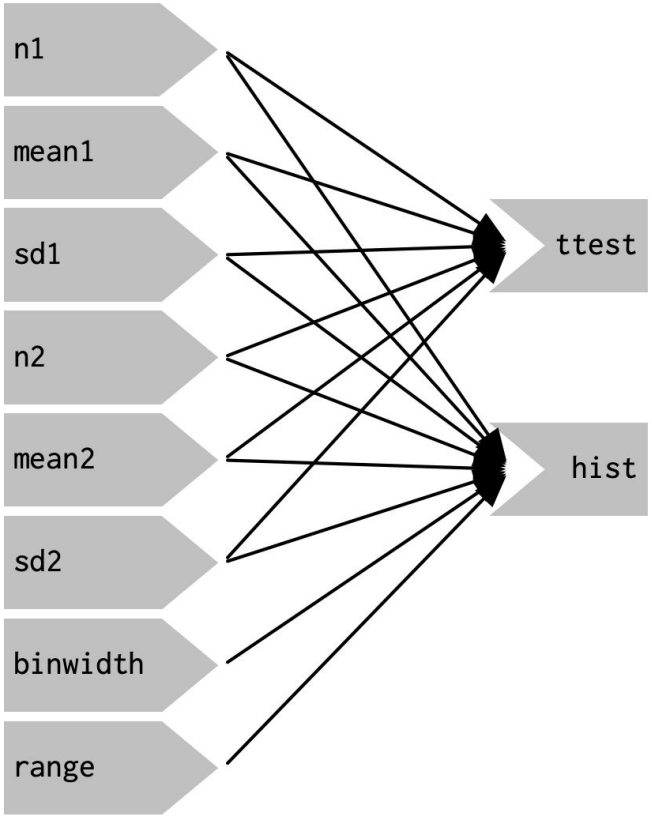
[to coding]

15 minute break

Be back by 7:32pm

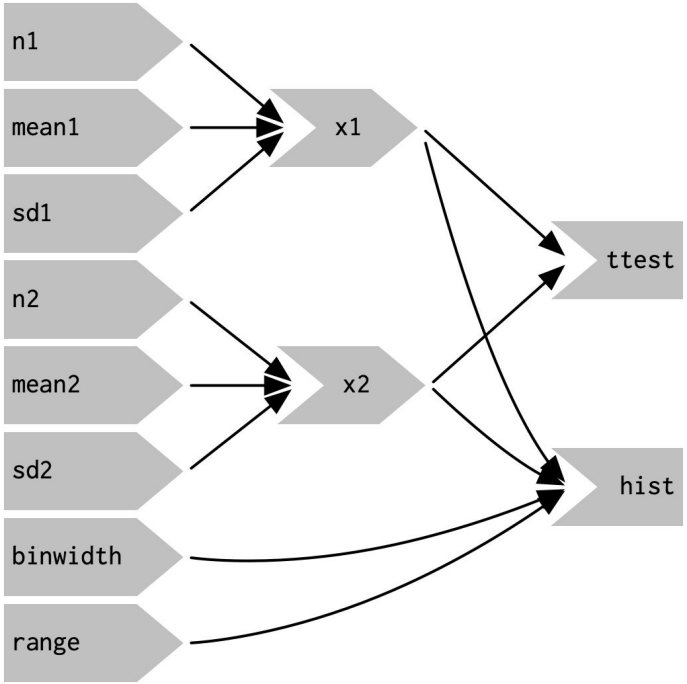
Purpose of Reactives

Without Reactives



Purpose of Reactives

With Reactives



Page Structure

Making App Easy to Read/Use

So far, we have relied on fluidRow() and column()

From last class, recall Tabs?

[to coding]

Layout Frameworks

The tabs Shiny app relied on a layout framework, which is a wrapper to make fluidRow and columns easier to specify

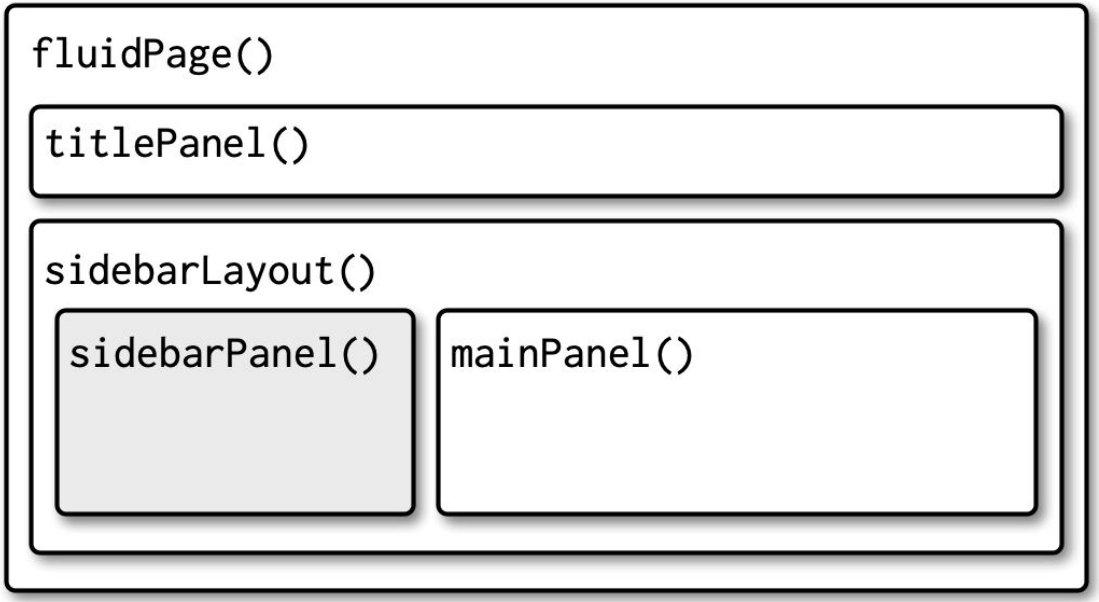
This class, we will focus on one popular layout framework: sidebarLayout

For a list of common frameworks, visit https://shiny.rstudio.com/articles/layout-guide.html

Making App Easy to Read/Use

Starter Kit

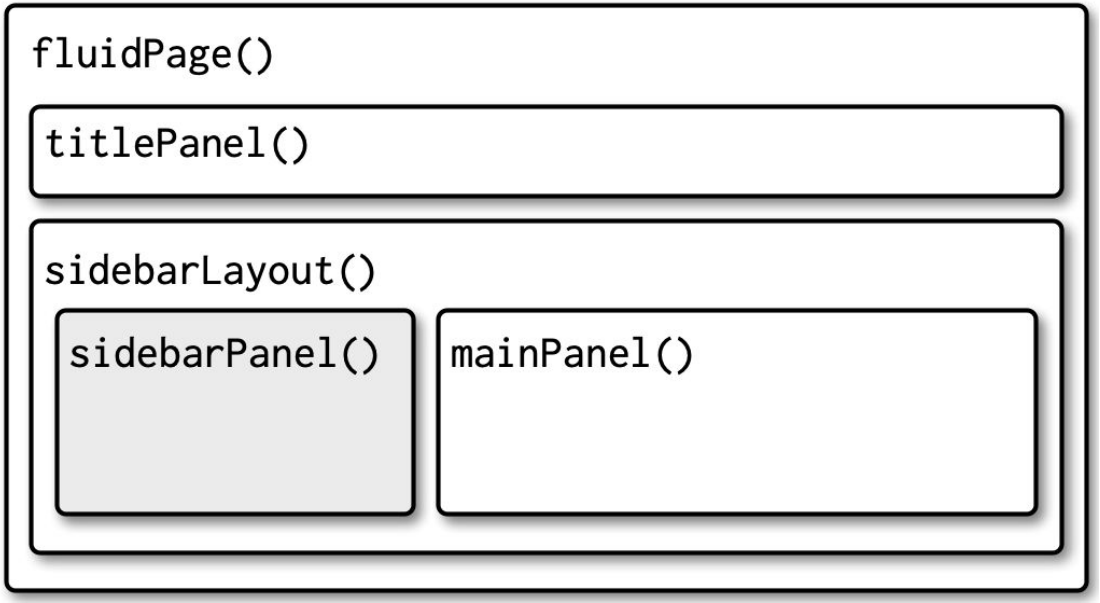
```
fluidPage(
  titlePanel(),
  sidebarLayout(
    sidebarPanel(),
    mainPanel()
  )
)
```



Making App Easy to Read/Use

Starter Kit

This is more than enough for your final project, but we will go over more powerful page organization functions next class

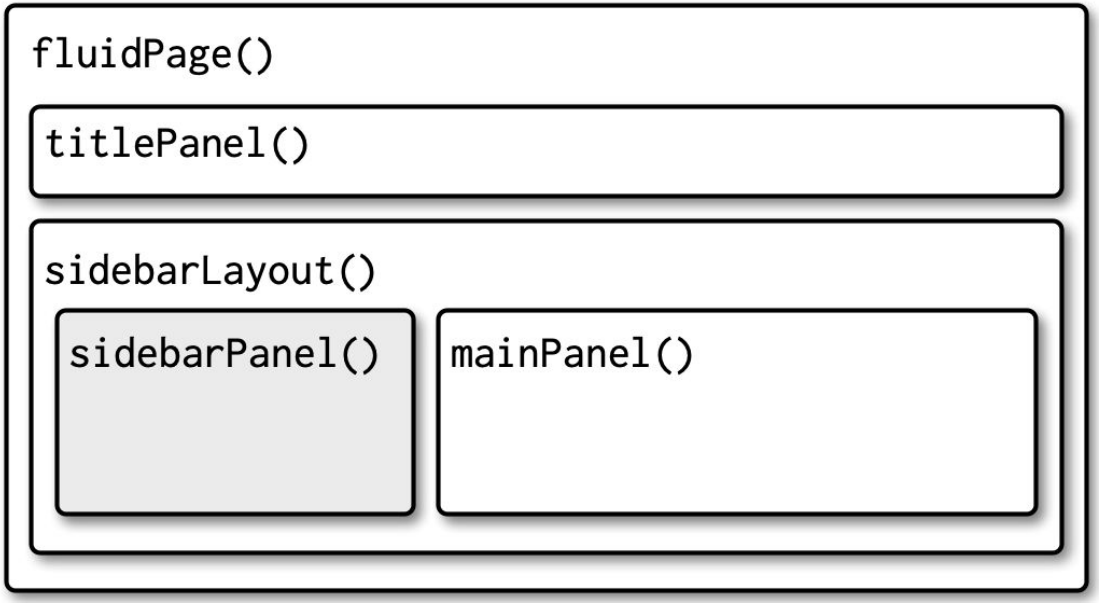


Making App Easy to Read/Use

Starter Kit

You specify this in the UI function

[to coding]



In Summary

Shiny Deep Dive

Basic building blocks of a data app:

- UI - interact with your users by taking their inputs and bringing them their outputs
- Server - the brain of your app, responsible for data wrangling and analysis
- Reactivity - improves the performance and UI of your app
- Page Structure - organizes your app

Making App Easy to Read/Use

```
fluidPage(
  titlePanel(),
  sidebarLayout(
    sidebarPanel(),
    mainPanel()
  )
)
```