

# What makes R Shiny so shiny?

A step-by-step introduction to interactive dashboards in R

Jean-Paul R. Soucy

Dalla Lana School of Public Health  
University of Toronto

June 5, 2021

# Learning objectives

After this workshop, you should be able to:

1. Understand the basic components of a Shiny app: ui and server
2. Use reactivity to create dynamic tables and plots
3. Setup a basic dashboard with multiple tabs of content
4. Get an idea of the tools available to extend the functionality of your dashboard

# Data and scripts for today's workshop

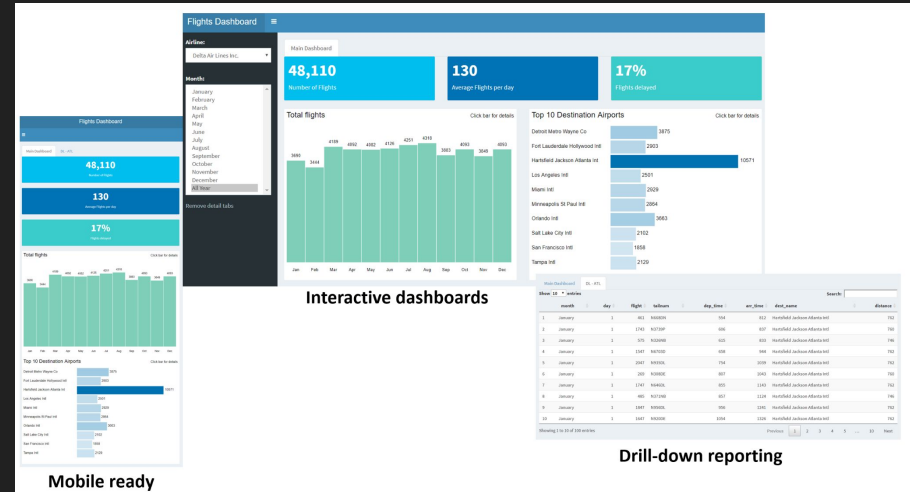
All R scripts used in this workshop are available in the following GitHub repository:

[tinyurl.com/cssc2021shiny](https://tinyurl.com/cssc2021shiny)

The R scripts and slides are available for asynchronous learning at your own pace.

# What is R Shiny?

Shiny: A framework for building interactive web apps in R

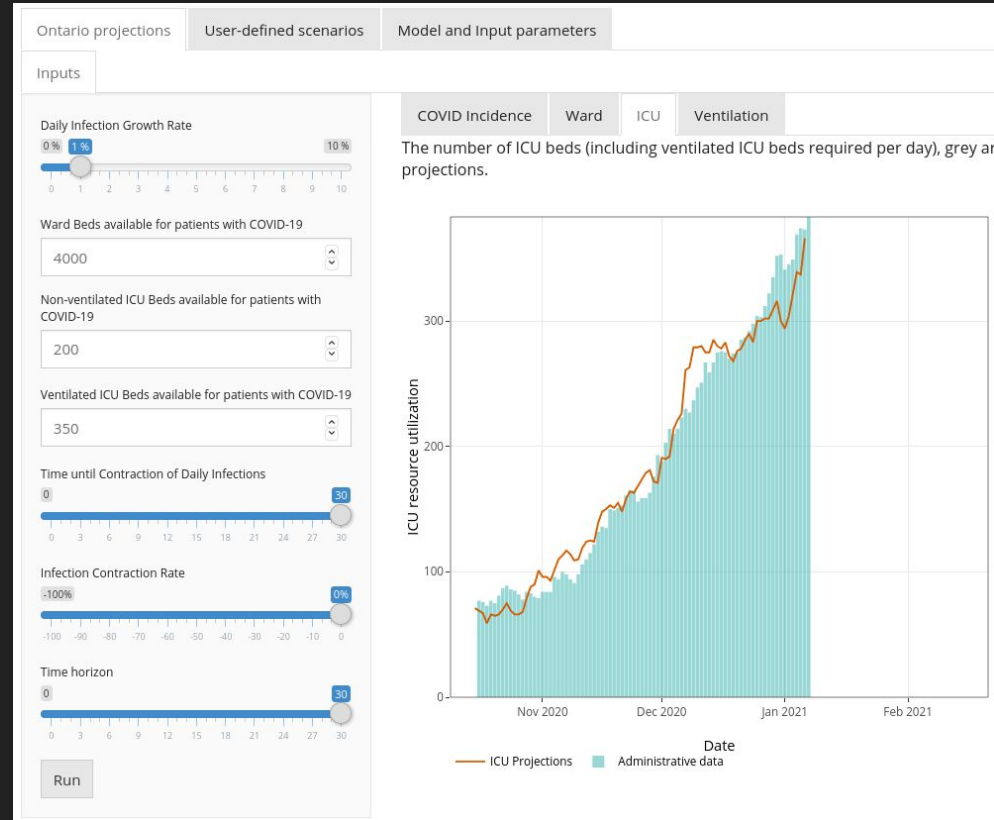


# Why make an interactive data visualization?

One use of interactive data visualization is to allow the user to test the assumptions of your model & adapt it to different scenarios.

One excellent example:

<https://www.covid-19-mc.ca/interactive-models/hospital-resources>



# IMDB dataset

We will be using two datasets derived from the IMDB dataset:

- All movies (horror and non-horror): average rating, total votes, runtime
- Horror movies: average rating, total votes, runtime, actor and director info

Raw dataset and data dictionary available:

<https://www.imdb.com/interfaces/>



# The two basic components of a Shiny app

## UI

- What your app looks like
- Basic building blocks are “Output” functions
- Also accepts raw HTML

## Server

- What your app does
- Basic building blocks are “render” functions
- Uses reactive programming

# Dashboard 1: Your first Shiny app

In dashboard 1, we will:

- Generate a functional, static webpage with two plots and some text
- Demonstrate basic usage of “ui” and “server”
- Use server to generate outputs that are displayed by ui



# UI: Basics

- `fluidPage` contains `fluidRow(s)`
- `fluidRow` contains 12 `column(s)`
- `column` contains your dashboard elements (text, plots, buttons, etc.)
- Defining all of these structures for every part of your Shiny app is not strictly necessary, but it does help maintain a consistent look across devices.

## Server: Basics

- Input: Access user-selected inputs (e.g., sliders, dropdown menu, text boxes)
- Output: Access objects generated in the server file (e.g., plots, text, UI elements)
- Session: Access information about the browser running the Shiny app

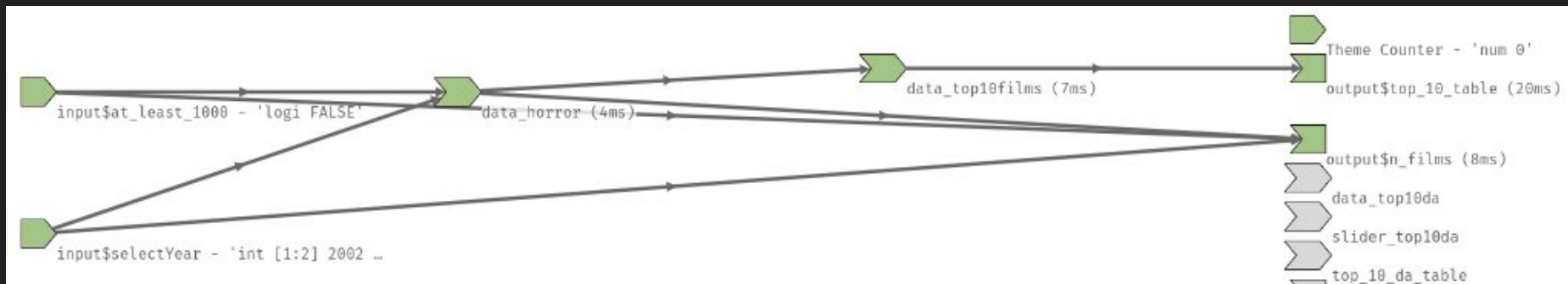
## Dashboard 2: Adding user input and reactivity

In dashboard 2, we will:

- Generate a functional, interactive webpage with a table that responds to user input
- Demonstrate basic usage “input” functions to accept user input
- Use these inputs inside the server function to generate a reactive “output” table that is displayed in ui

# Reactivity

- Under the hood, Shiny uses a “reactive graph” to invalidate and update outputs in response to user inputs
- Shiny can generate a “reactlog” for you and show step-by-step how the underlying variables of your dashboard change in response to user input



## Dashboard 3: shinydashboard

In dashboard 3, we will:

- Use the “shinydashboard” package to convert dashboard 2 into a proper Shiny dashboard (header, body, sidebar)
- Use the “dashboardthemes” package to customize the appearance of our dashboard

## Dashboard 4: Tabs and tabsets

In dashboard 4, we will:

- Build on dashboard 3 through the addition of multiple “tabs” of content accessible through the dashboard sidebar
- Demonstrate how to make multiple “panels” of content available on a single tab through the use of “tabset panels”

# Tabs and tabsets

**Tabs** allow you to have multiple pages of content in your dashboard, accessible from the sidebar. Tabs are defined in two places:

- Sidebar: sidebarMenu contains menuItem(s)
- Body: tabItems contain tabItem(s), which define actual content

**Tabset panels** allow you to have multiple “panels” of content on the same page. Tabset panels are defined within fluidPage(s):

- tabsetPanel contains tabPanel(s)

## Dashboard 5: Extending your R Shiny dashboard

In dashboard 5, we will:

- Show how various R packages can be used to seamlessly integrate with R Shiny
- Demonstrate the use of the JavaScript library “plotly”

Check out the examples available in htmlwidgets for R:

<http://www.htmlwidgets.org/>



# Common mistakes

Avoid the following common mistakes, which can cause your Shiny app to fail silently:

- Never duplicate IDs (e.g., you cannot use `plotOutput("plot_1")` in two places, even if you want to show the same plot on different pages)
- Render functions go in server and must be assigned to output (e.g., `output$plot_id <- renderPlot(plot(y~x))`)
- Output functions go in ui (e.g., `plotOutput("plot_id")`)
- Double check that IDs match exactly in render and output functions

# Best practices

## Organizing your code:

- global.R (runs first; objects accessible by both ui and server)
- ui.R (define how your app looks)
- server.R (define what your app does)
- app.R (use “source” to read in the above and launch app)

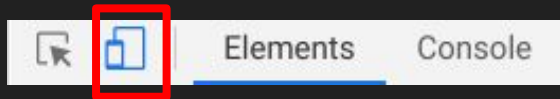
## Optimizing for mobile:

- Use “device toolbar” in Chrome/Chromium browser
- Visit local host:

Listening on http://127.0.0.1:6124

127.0.0.1:6124

- F12 →



# Acknowledgements and additional resources

- Mastering Shiny (Hadley Wickham):  
<https://mastering-shiny.org/index.html>
- Analyzing IMDb Data The Intended Way, with R and ggplot2 (Max Woolf): <https://minimaxir.com/2018/07/imdb-data-analysis/>
- Shiny code examples: <https://shiny.rstudio.com/gallery/>
- My GitHub has several Shiny dashboards:  
<https://github.com/jeanpaulrsoucy>
- Short-term Mortality Fluctuations [English, code available]:  
<https://mpidr.shinyapps.io/stmortality/>
- Real-time mortality (Portugal) [Portuguese]:  
<https://evm.min-saude.pt/>

# Alternatives to R Shiny

## Open source alternatives

- R Markdown
- flexdashboard (R)
- d3 (JS)

## Proprietary alternatives

- Tableau
- Power BI
- IBM Cognos

# Jean-Paul R. Soucy

Personal website: [jprs.me](http://jprs.me)

Twitter: [@JPSoucy](https://twitter.com/JPSoucy)

My research is focused on infectious disease epidemiology, particularly **COVID-19**, **antimicrobial resistance** and the use of **emerging data sources** for infectious disease surveillance.

Please feel free to reach out to me.

