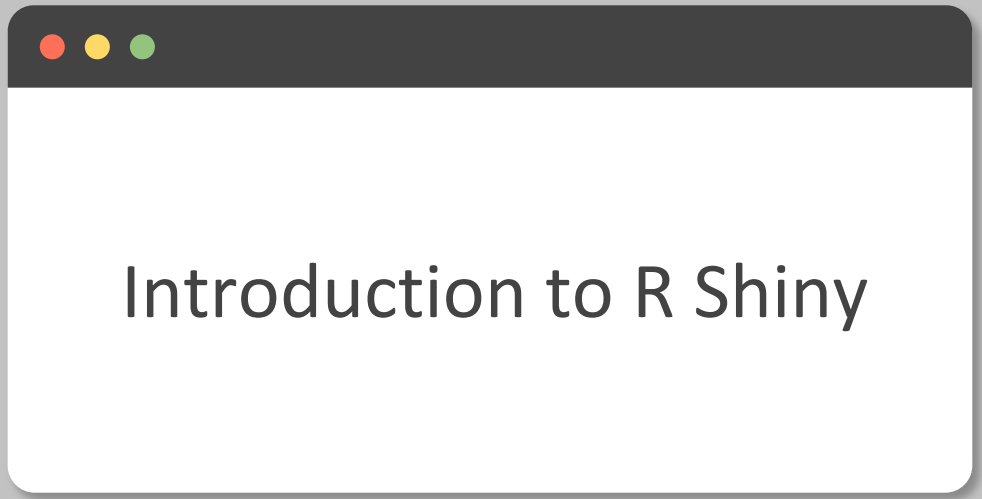
Workshop Series:
Reusable Research Data Made Shiny

Ontario Dairy Research Centre | Online
February 21st - 24th, 2023

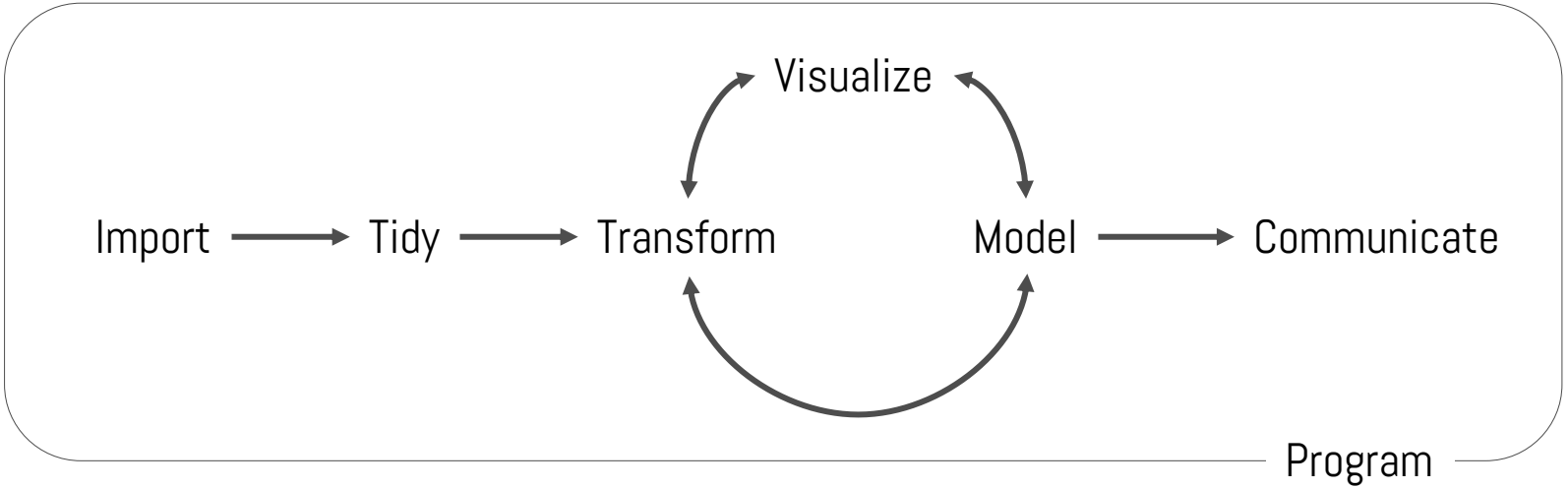
University of Guelph
Improve Life.

Compute Ontario

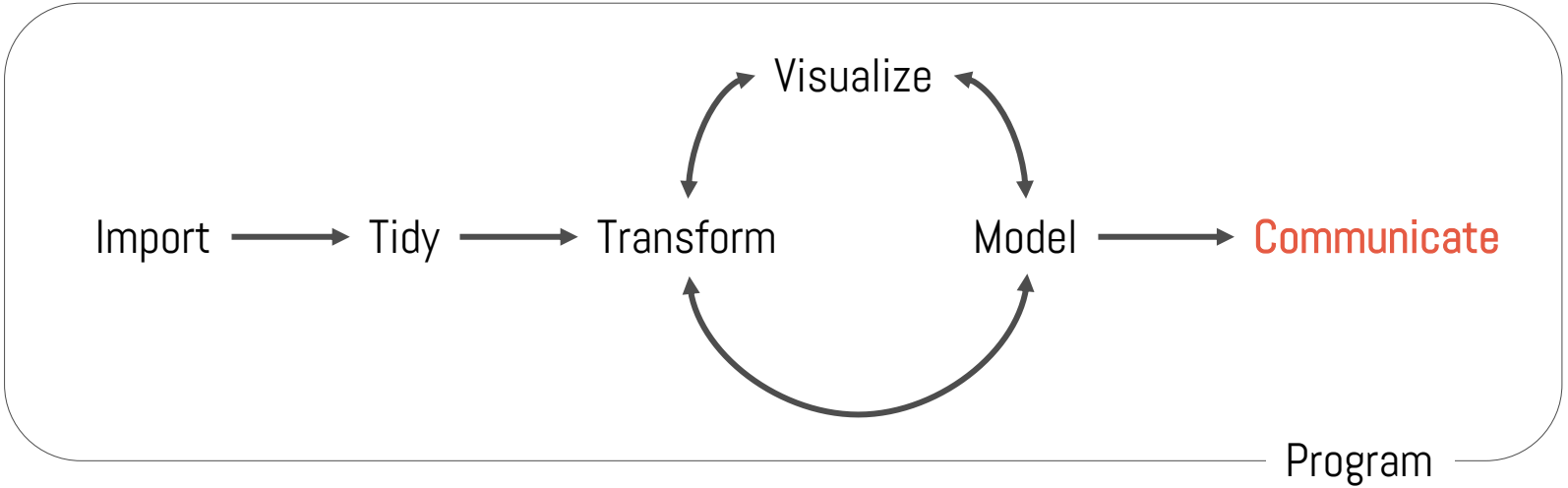
ADC


Introduction to R Shiny

Introduction to R Shiny



Introduction to R Shiny



Welcome to Shiny!

Shiny is an R package that makes it easy to build interactive web applications straight from R.
No web development skills are required

To share your Shiny app, you can:

- Host standalone apps on a webpage
- Embed them in R Markdown documents
- Build dashboards

You can also extend your Shiny apps with:

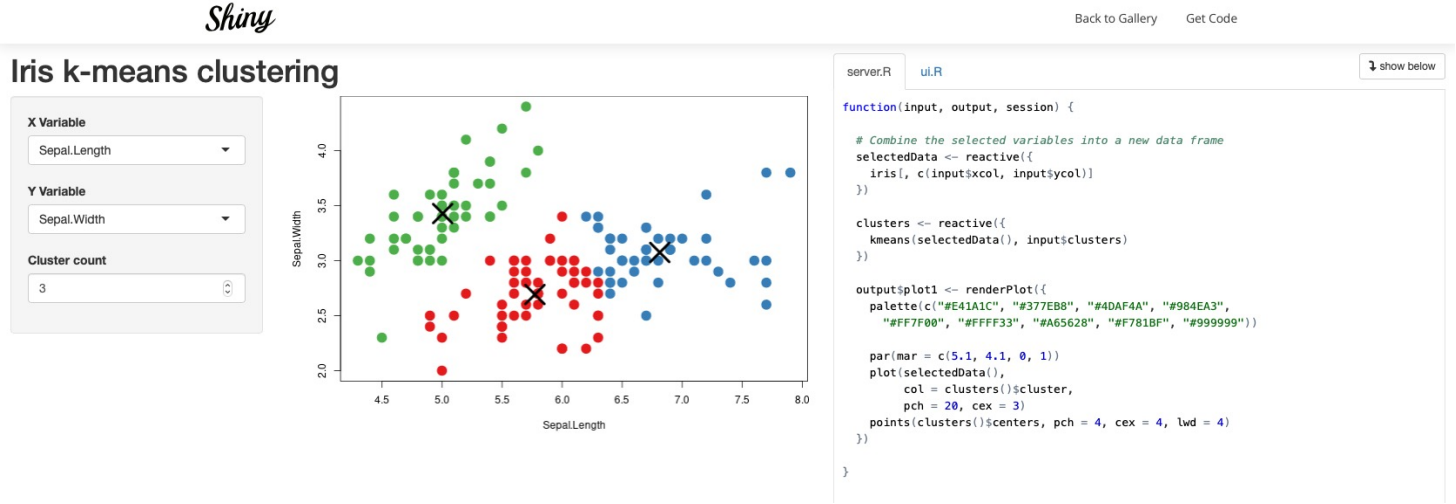
- CSS themes
- HTML Widgets
- JavaScript actions

Adapted from: https://shiny.rstudio.com/articles and https://mastering-shiny.org

Welcome to Shiny!

Let's check the "Iris k-means clustering" app from the official Shiny App gallery:

https://shiny.rstudio.com/gallery/kmeans-example.html



Welcome to Shiny!

global.R – Has everything that needs to be ready before the app starts

- Executed only **once before** the app starts
- Can be accessed from both *ui* and *server* sides

ui.R – Controls the layout and appearance of your app (front end)

- Executed only **once when** the app starts

server.R – Controls the logics of your app

- Executed **as many times as** you program it to (back end)

Welcome to Shiny!

Let's (re)do this app on Posit Cloud

Welcome to Shiny!

What is on ui.R that could go to global.R?

How about on server.R? Anything there?

What happens if I move line 17 from ui.R to the end of ui.R?

- line 17: vars <- setdiff(names(iris), "Species")


User Interface (UI)

User Interface (UI)

There are several packages available to customize your Shiny App. You can find a curated list of resources here: https://github.com/nanxstats/awesome-shiny-extensions#ui-components

Input functions examples:

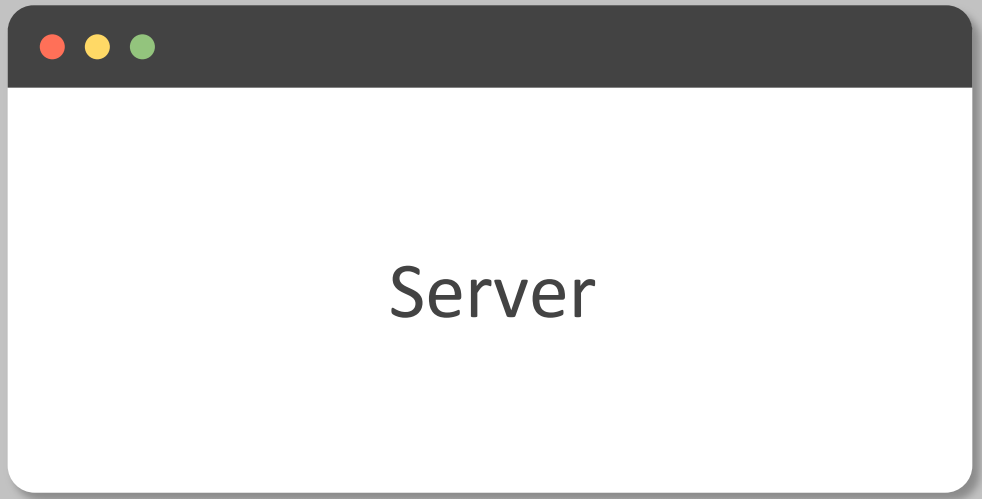
- sliderInput()
- selectInput()
- textInput()
- numericInput()

They all take different arguments, but the first one is always the same: **inputId**

- Must be unique across your app
- Must have only letters, numbers, and underscores

Your turn!

Modify the k-means app to change the numericInput to a sliderInput. Then, add animation to the input widget so when the user presses play the input widget scrolls through the range automatically.


Server

Server

input

- It is a list-like object that contains all the input data sent from the browser, named according to the input ID
- It's read-only (can't be modified after UI is rendered*)
- Can only be read inside a **reactive context** (e.g., render or reactive functions)

output

- It is very similar to input: it's also a list-like object named according to the output ID. The main difference is that you use it for sending output instead of receiving input
- You always use the output object in concert with a render function
- You can't read an output object

Adapted from: https://shiny.rstudio.com/articles and https://mastering-shiny.org

Your turn!

Given this UI:

```
ui <- fluidPage(
  textInput("name", "What's your name?"),
  textOutput("greeting")
)
```

** Fix and run them to make sure they work properly

What is wrong in each of these server functions?

```
server1 <- function(input, output, server) {
  input$greeting <- renderText(paste0("Hello ", name))
}
```

```
server2 <- function(input, output, server) {
  greeting <- paste0("Hello ", input$name)
  output$greeting <- renderText(greeting)
}
```

```
server3 <- function(input, output, server) {
  output$greting <- paste0("Hello", input$name)
}
```

Your turn!

Given this UI:

```
ui <- fluidPage(
  textInput("name", "What's your name?"),
  textOutput("greeting")
)
```

** Fix and run them to make sure they work properly

What is wrong in each of these server functions?

```
server1 <- function(input, output, server) {
  input$greeting <- renderText(paste0("Hello ", name))
}
```

```
server2 <- function(input, output, server) {
  greeting <- paste0("Hello ", input$name)
  output$greeting <- renderText(greeting)
}
```

```
server3 <- function(input, output, server) {
  output$greting <- paste0("Hello", input$name)
}
```

Wrap-up!

Today we talked about

1. Tidy Data with Tidyverse
    1. Filter
    2. Select
    3. Arrange
    4. Summarize
    5. Group By
    6. Mutate
    7. Pivot Longer
    8. Pivot Wider
2. Introduction to the Shiny Framework
    1. UI
    2. Server
    3. Global

Wrap-up!

Tomorrow we will talk about

1. App reactivity
2. App layout
3. Building you own App
4. Publish your App online