

## Shiny - Part 2

- ▶ Main Components of a Shiny Web App
- ▶ Shiny programs are the easiest to build and understand using two scripts, which are kept within the same folder. They should be named `server.R` and `ui.R`.
- ▶ The input elements are defined in `ui.R` and processed by `server.R`, which then sends them back to `ui.R`
- ▶ Consideration: Reactive Programming

# Shiny Basics

## Basic structure of a Shiny program

- ▶ Selection of simple input widgets (checkboxes and radio buttons)
- ▶ Selection of simple output types (rendering plots and returning text)
- ▶ Selection of simple layout types (page with sidebar and tabbed output panel)
- ▶ Handling reactivity in Shiny

## Running a Shiny App

To run a Shiny program on your local machine you just need to do the following:

1. Make sure that `server.R` and `ui.R` are in the same folder.
2. Make this the R's working directory (using the `setwd()` command, for example `setwd("/shinyFiles/minimalExample")`).
3. Load the Shiny package (`library(shiny)`). You should always do that in both `server.R` and `ui.R` files.
4. Type `runApp()` at the console.

## runApp

- ▶ `runApp()` with the name of a directory within works just as well, for example, `runApp(" /shinyFiles/minimalExample")`.
- ▶ Just remember that it is a directory and not a file that you need to point to.

## ui.R

- ▶ The ui.R file is a description of the UI and is often the shortest and simplest part of a Shiny application.
- ▶ All of the UI elements are defined within this instruction.
- ▶ The standard shiny layout is a three panel layout, with a header panel, a sidepanel controls on the left, and the main panel on the right - with the output.
- ▶ This layout is called **pageWithSidebar**. There are other layouts too - such as basicPage and threePage.

# Inputs

The arguments are pretty typical among most of the widgets and are as follows:

**inputId** : This argument names the variable so it can be referred to in the server.R file

**label** : This argument gives a label to attach to the input so users know what it does

**value** : This argument gives the initial value to the widget when it is set up. All the widgets should have sensible defaults for this argument.

# Main Panel

- ▶ The final function is `mainPanel()`, which sets up the output window.
- ▶ HTML helper functions to make a little title `h3("...")`.  
Knowledge of HTML is very useful.
- ▶ There are several of these functions designed to generate HTML to go straight on the page; type `?p` at the console for the complete list.

## Main Panel

- ▶ The other element that goes in `mainPanel()` is an area for handling reactive text or plots generated within the `server.R` file
- ▶ For example - a call to `textOutput()` with the name of the output as defined in `server.R`, in the upcoming "minimal case", "textDisplay".



- ▶ `shinyServer(.....)` defines the bit of Shiny that's going to handle all the data.
- ▶ On the whole, two types of things go in here. Reactive objects (for example, data) are defined, which are then passed around as needed (for example, to different output instructions), and outputs are defined, such as graphs

# Reactive Programming

## Simple R example

```
> A <- 5
> B <- A + 3
> A <-6           #Update A
>
> c(A,B,A+3)
[1] 6 8 9
>
```