THE UNIVERSITY OF ARIZONA
Mel & Enid Zuckerman
College of Public Health

Interactive Documents and Applications with R

Dominic LaRoche

October 18, 2015

Outline

- Introduction
- Example document and example application
- Nuts and Bolts

Section 1

Introduction

What are interactive documents and applications?

Interactive documents and applications are graphical user interfaces which run analyses in the background.

- Allow user to interact with analyses
- Run pre-defined analyses
- Provide results

How are interactive documents and applications beneficial?

Interactive Applications:

- Streamline repetetive analyses and reduce errors

How are interactive documents and applications beneficial?

Interactive Applications:

- Streamline repetetive analyses and reduce errors
- Increase efficiency of an organization by enabling others to perform and interpret simple analyses

How are interactive documents and applications beneficial?

Interactive Applications:

- Streamline repetetive analyses and reduce errors
- Increase efficiency of an organization by enabling others to perform and interpret simple analyses
- Provide a high-value service to customers

How are interactive documents and applications beneficial?

Interactive Applications:

- Streamline repetetive analyses and reduce errors
- Increase efficiency of an organization by enabling others to perform and interpret simple analyses
- Provide a high-value service to customers

Interactive Documents:

- Provide a tool to both inform *and* educate

How are interactive documents and applications beneficial?

Interactive Applications:

- Streamline repetetive analyses and reduce errors
- Increase efficiency of an organization by enabling others to perform and interpret simple analyses
- Provide a high-value service to customers

Interactive Documents:

- Provide a tool to both inform *and* educate
- Allow users to ask and answer their own questions
  - Facilitate discovery
  - Understand the "why" and not just the "what"

How are interactive documents and applications beneficial?

Other potential uses:

- Teaching

How are interactive documents and applications beneficial?

Other potential uses:

- Teaching
- Marketing

How are interactive documents and applications beneficial?

Other potential uses:

- Teaching
- Marketing
- Public health

Interactive Applications

Written in R:

- Can do anything you can do in R
- Large library of pre-written widgets for user interaction
- Can accomodate anything you can write in HTML, Java, Python, etc.

- Can be hosted locally and run behind a firewall
- Use shiny.io for free (up to 5 apps)

Interactive Reports

Written in Rmarkdown:

- Combines report text and code into a single document
- Can do anything you can do in R
- Felixble formatting
- Can easily convert between document types
- Can create a static document from interactive one
- Document must be hosted

Interactive Reports

Written in Rmarkdown:

- Combines report text and code into a single document
- Can do anything you can do in R
- Felixble formatting
- Can easily convert between document types
- Can create a static document from interactive one
- Document must be hosted
  - On the web
  - Locally as application

Section 2

Examples

Examples

- Interactive Application
- Interactive Document
- Shiny Example Library

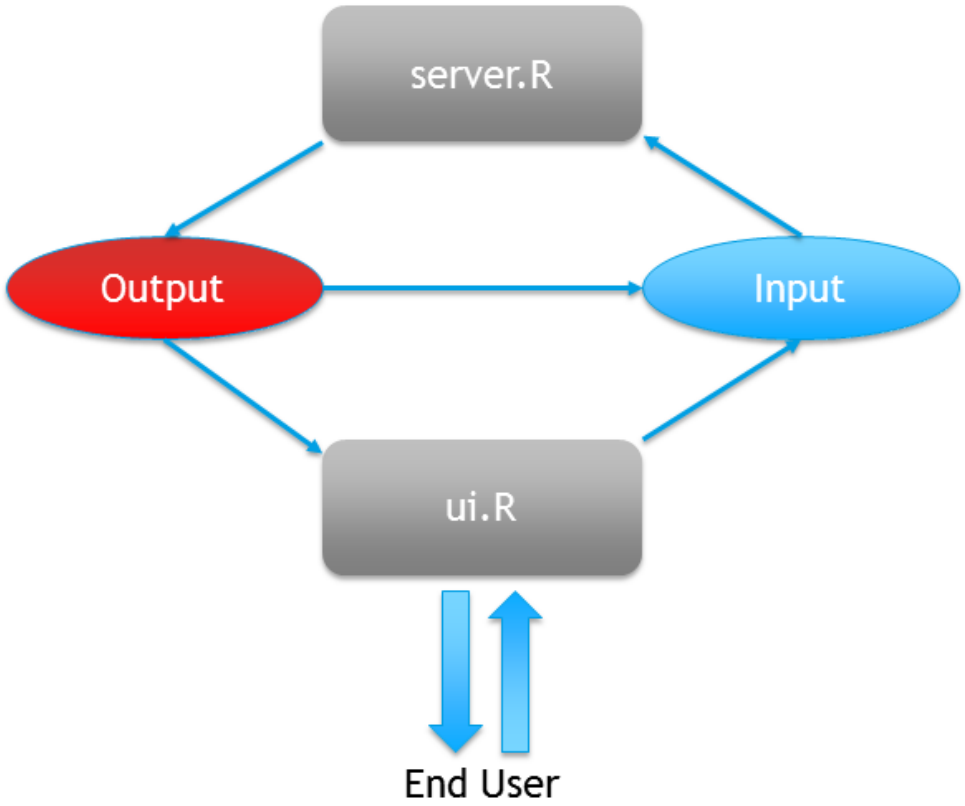
Section 3

Nuts and Bolts

WARNING!

I will be discussing some details of creating these type of applications and documents which will involve talking about code and programming.
Feel free to escape!

Architecture of an Application



Building an Application

Applications depend on two main objects:

- input
  - Stores all the user inputs for use in the analysis
  - Data, parameters, etc.

Building an Application

Applications depend on two main objects:

- input
  - Stores all the user inputs for use in the analysis
  - Data, parameters, etc.

- output
  - Stores all of the outputs from the analysis
  - Everything returned to the user

Building an Application

Applications use two main functions from the shiny package:

- ShinyServer
  - Controls the server logic
    - What happens and in what order
  - Calls sub-functions or other programs
  - Creates dynamic inputs
  - uses "inputs" to create "outputs"

Building an Application

Applications use two main functions from the shiny package:

- ShinyServer
  - Controls the server logic
    - What happens and in what order
  - Calls sub-functions or other programs
  - Creates dynamic inputs
  - uses "inputs" to create "outputs"

- ShinyUI
  - Defines the appearance of the application
  - Interacts with the user to collect "inputs"
  - Displays all the "outputs" of the analysis

Pre-packaged Inputs

Pre-pacakged Outputs

Reactivity

The analysis must respond to updated user inputs. This is known as **reactivity**.

- 3 main components for reactivity
  1.) Reactive source (generally an input)
  2.) Reactive conductor (usually a function)*
  3.) Reactive endpoint (usually an output)

- Trickiest part for typical R programmer

- All manipulations of reactive values must be done inside the reactive environment

Reactivity

Some of the tricky things about reactivity:

- A reactive function will not execute if there is no reactive endpoint

Reactivity

Some of the tricky things about reactivity:

- A reactive function will not execute if there is no reactive endpoint
- A reactive value is always reacting to user inputs
  - Some inputs may have no initial value
  - Other inputs may require typing

Reactivity

Some of the tricky things about reactivity:

- A reactive function will not execute if there is no reactive endpoint
- A reactive value is always reacting to user inputs
  - Some inputs may have no initial value
  - Other inputs may require typing
- Can use observer() function as an endpoint if no output is required

Reactivity

Some of the tricky things about reactivity:

- A reactive function will not execute if there is no reactive endpoint
- A reactive value is always reacting to user inputs
  - Some inputs may have no initial value
  - Other inputs may require typing
- Can use observer() function as an endpoint if no output is required
- isolate() function + action button can control unwanted reactivity

Reactivity

Some of the tricky things about reactivity:

- A reactive function will not execute if there is no reactive endpoint
- A reactive value is always reacting to user inputs
  - Some inputs may have no initial value
  - Other inputs may require typing
- Can use observer() function as an endpoint if no output is required
- isolate() function + action button can control unwanted reactivity
- validate() function can require certain conditions are met before reacting

Dynamic Applications

Extensive applications should be dynamic

- Limit the number of inputs to focus the user

Dynamic Applications

Extensive applications should be dynamic

- Limit the number of inputs to focus the user
- Ask for inputs as needed and build up options

Dynamic Applications

Extensive applications should be dynamic

- Limit the number of inputs to focus the user
- Ask for inputs as needed and build up options
- Make inputs from reactive outputs

Building an Interactive Document

Interactive document is a special case of an application

Building an Interactive Document

Interactive document is a special case of an application

- Written in Rmarkdown
  - Markup language (think LaTeX) that weaves code and text into single document

Building an Interactive Document

Interactive document is a special case of an application

- Written in Rmarkdown
  - Markup language (think LaTeX) that weaves code and text into single document

- Output controlled by header: word, pdf, latex presentation, io slides, and HTML

Building an Interactive Document

Interactive document is a special case of an application

- Written in Rmarkdown
  - Markup language (think LaTeX) that weaves code and text into single document
- Output controlled by header: word, pdf, latex presentation, io slides, and HTML
- Specify HTML with shiny runtime to get interactive document.

Building an Interactive Document

Some differences from a standard application:

Building an Interactive Document

Some differences from a standard application:

- Only 1 file (no separate server.R and ui.R files)

Building an Interactive Document

Some differences from a standard application:

- Only 1 file (no separate server.R and ui.R files)
- Use inputPanel instead of shinyUI() function

Building an Interactive Document

Some differences from a standard application:

- Only 1 file (no separate server.R and ui.R files)
- Use inputPanel instead of shinyUI() function
- Outputs are rendered in 2 stages:
  1.) Rendered into HTML by shiny package
  2.) Placed into document via Rmarkdown arguments

Questions?

"No mom, we didn't do our abc's today, we just reviewed our R, Rmarkdown, and HTML...