Shiny in Bioinformatics

RStudio, Inc.
July 19, 2013
BioC Conference
Shiny Introduction

Joe Cheng
RStudio, Inc.
R is great!

- Excellent for statistics
- Amazing at data visualization
- Vibrant and generous community
...but...

- A personal experience, not a shared one
- Sharing results generally means exporting to a static format
- Need to write code to manipulate visualizations and other output
- Modern visualization is animated and interactive - not obvious how to do this from R
Shiny is
R + interactivity + web
made easy
What is Shiny?

Hello Shiny!

Number of observations:
0 500 1,000

Histogram of dist

Frequency

-3 -2 -1 0 1 2 3
dist
What is Shiny?

- Interactive web application around your R analyses
- Zero HTML/CSS/JavaScript knowledge required...
- … but fully customizable and extensible with HTML/CSS/JavaScript
What is Shiny?

- Modern web UI with attractive defaults
- Designed to integrate with existing JavaScript libraries (including d3.js)
- Uses a “reactive” programming model which allows dramatically simpler code than traditional UI or web programming
What is Shiny?

- Free and open source (GPLv3) R package
- Created and maintained by RStudio, Inc.
- First released in November 2012
Deploying Shiny Apps

- R package itself is not designed for deployment
- Shiny Server (AGPLv3)
  - [https://github.com/rstudio/shiny-server](https://github.com/rstudio/shiny-server)
- Shiny Server “Pro” (Commercial)
  - Adds authentication, security, monitoring features to Shiny Server
  - Coming soon
- Shiny hosting on rstudio.com
  - Currently in free beta; sign up at [http://rstudio.com/shiny](http://rstudio.com/shiny)
Shiny in Action
Visual Models

http://spark.rstudio.com/genentech/generic-response-prediction-tool-web-application-demonstration-v2/
Interactive Plotting

[Image of a computer screen showing a software interface for reconstructing gene networks.]

http://glimmer.rstudio.com/qbrc/grn/
Interactive Plotting

http://glimmer.rstudio.com/qbrc/heatmap/
Analytics That Go Viral

http://spark.rstudio.com/jkatz/SurveyMaps/
http://www.huffingtonpost.com/2013/06/06/dialect-maps_n_3395819.html
Build Practical Tools

http://glimmer.rstudio.com/systematicin/retirement.withdrawal/
Customizable Dashboards

http://glimmer rstudio.com/winstontest/shiny-jsdemo/
Control JavaScript Visualizations

http://glimmer.rstudio.com/jcheng/leaflet-demo/
displayR

Shawn Balcome
University of Minnesota
Shiny Walkthrough

Jeff Allen
RStudio, Inc.
# Update Shiny
install.packages("devtools")
library(devtools)
install_github("shiny", "rstudio")

# Download Examples
Shiny Examples

- Data from `curatedOvarianData`
- 100s of MB of microarray data
- We extract two matrices:
  - 8 Normal samples x 8 genes
  - 20 Tumor samples x 8 genes
- Store as ExpressionSet objects
- Save as .Rds files
Example 1

- Comparison of normal expression vs. tumor expression
- Allow user to select gene to visualize
- Render as boxplot

http://bit.ly/biocon1
library(shiny)
options(shiny.launch.browser=FALSE)

tumor
normal
library(Biobase)
expres(tumor)
boxplot(list(Tumor=expres(tumor), Normal=expres(normal)))
Let's Do it!
Dependency Graph - Example 1

```
output$genePlot

renderPlot

input$gene
```
Reactivity

- Reactive programming is a way of organizing your code that emphasizes values that can change over time...

- …and expressions that depend on such values...

- …so that changes to those values naturally “flow” through your system.
Reactivity Demo
Reactivity

- **Reactive values** are values that vary over time

- **Reactive expressions** are expressions that may depend on reactive values and other reactive expressions

Shiny Examples, Cont'd.

Jeff Allen
RStudio, Inc.
Shiny Example 2

- Shiny offers text-based inputs and outputs
- View tables
- Print R output directly
- Let's create an app that can preview the gene expression data in each set

Dependency Graph - Example 2

output$ssummary → renderPrint

input$cols

output$view → renderTable

input$type

input$rows
Dependency Graph - Example 2

- `output$summary` -> `renderPrint`
- `input$cols` -> `renderPrint`
- `input$type` -> `renderTable`
- `output$view` -> `renderTable`
- `input$rows` -> `renderTable`
- `expensive`
Dependency Graph - Example 2

- `output$summary` depends on `renderPrint`
- `input$cols` depends on `renderPrint`
- `input$type` depends on `renderPrint`
- `output$view` depends on `renderTable`
- `input$rows` depends on `renderTable`
Dependency Graph - Example 2

- `output$summary` -> `renderPrint`
- `input$cols` -> `renderPrint`
- `input$type` -> `renderPrint`
- `output$view` -> `renderTable`
- `datasetInput` -> `renderPrint`
- `input$rows` -> `renderPrint`
Dependency Graph - Example 2

- output$summary
- renderPrint
- input$type
- input$cols
- datasetInput
- output$view
- renderTable
- input$rows
- expensive
Shiny Example 3

- No limits to computational power of Shiny
- Anything you can do in R, you can do in server.R

Shiny Example 4

- Shiny not only used in isolation
- Great for injecting some interactivity at key points in a larger analysis workflow
- Use `stopApp()` to return some data from a Shiny app

Next Steps

- **Shiny homepage**
  [http://rstudio.com/shiny](http://rstudio.com/shiny)

- **Shiny Tutorial**

- **Discussion list**
  [http://groups.google.com/group/shiny-discuss](http://groups.google.com/group/shiny-discuss)
Thanks!

Q&A