Dynamic dashboards with Shiny

Winston Chang
@winston_chang
RStudio

April 2015

Overview

- What is a data dashboard?
- A bit about Shiny
- Using **shinydashboard** package
- Deploying a dashboard

Dashboards

shinyapps.io

Help | Account: winston | Winston Chang

- Dashboard
- Applications
  - All
  - Running
  - Sleeping
  - Archived
- Account

APPLICATION 10620 – GGVIS-BASICS-SLIDER

Overview | Metrics | Settings | Users | Logs | Restart | Rebuild | Archive

OVERVIEW

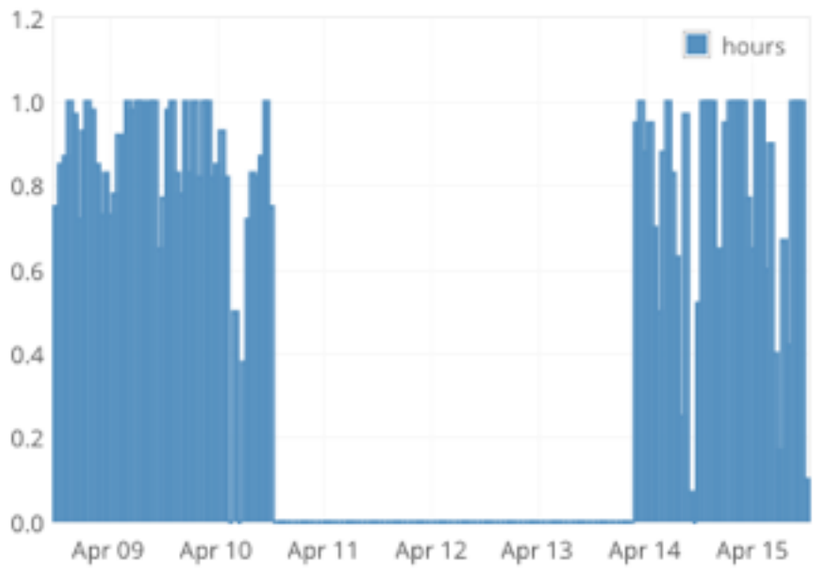
Id	10620
Name	ggvis-basics-slider
URL	http://winston.shinyapps.io/ggvis-basics-slider
Status	Running
Size	small
Deployed	Dec 4, 2014
Updated	Apr 15, 2015
Created	Feb 24, 2014
Bundle	Download

INSTANCES

Id: 92966

APPLICATION USAGE

Total: 69.42 hours



What does a dashboard do?

- Convey information efficiently
- Provide intuitive user interface
- Look attractive
- Spectrum, from presentation-focused to exploration-focused

How does a dashboard work?

1. Fetch data
2. Process/summarize the data
3. Concisely present processed data
4. (Optional) Provide exploration tools

Fetching data

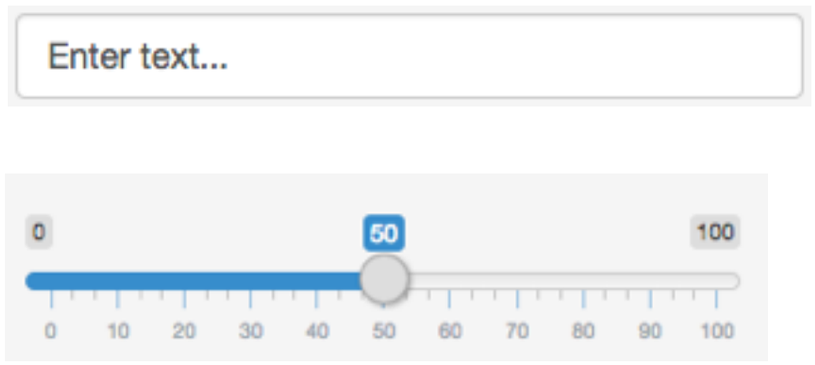
- Must be quick
- Database (DBI package)
- Scheduled data dump/summary (CSV + others)
- Web API (jsonlite, XML packages)

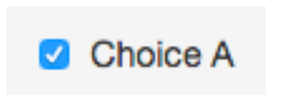
Presenting data

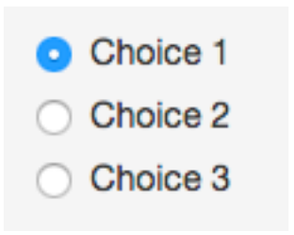
- Numbers & text (`renderText`)
- Tables (`renderTable`, `DT` package)
- Graphs (`renderPlot`)

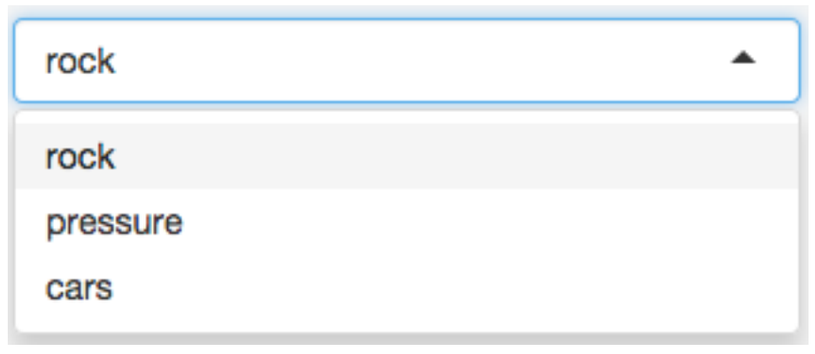
Components for exploration

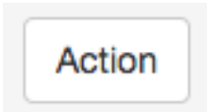
User inputs in Shiny











New interactive components

- DT: interactive tables
- leaflet: interactive maps
- Interactions with base R graphics and ggplot2

Shiny basics

What is Shiny?

- R package
- Platform for creating web apps in R
- Uses a reactive programming model
- Free software (GPL v3)

```
library(shiny)

ui <- basicPage(
  sliderInput("n", "Number of points:",
              min = 10, max = 500, value = 100),
  plotOutput("distPlot")
)

server <- function(input, output) {
  output$distPlot <- renderPlot({
    plot(rnorm(input$n), rnorm(input$n))
  })
}

shinyApp(ui, server)
```

```
library(shiny)

ui <- basicPage(
  sliderInput("n", "Number of points:",
              min = 10, max = 500, value = 100),
  plotOutput("distPlot")
)

server <- function(input, output) {
  output$distPlot <- renderPlot({
    plot(rnorm(input$n), rnorm(input$n))
  })
}

shinyApp(ui, server)
```

Using shinydashboard

What is shinydashboard?

- R package for creating dashboard-style layouts with Shiny

- Shiny uses Bootstrap for layout
- Shinydashboard uses AdminLTE, which a theme built on top of Bootstrap

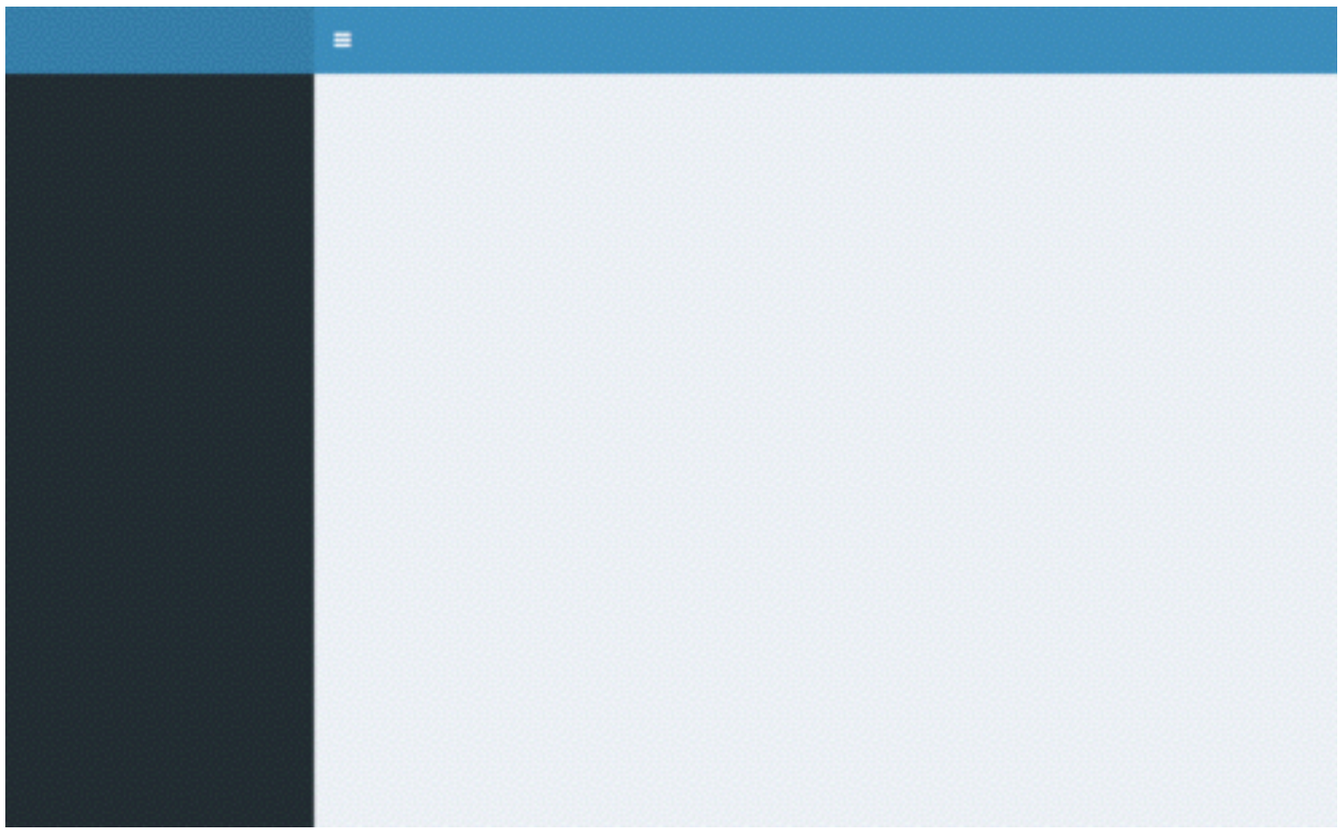
Installation

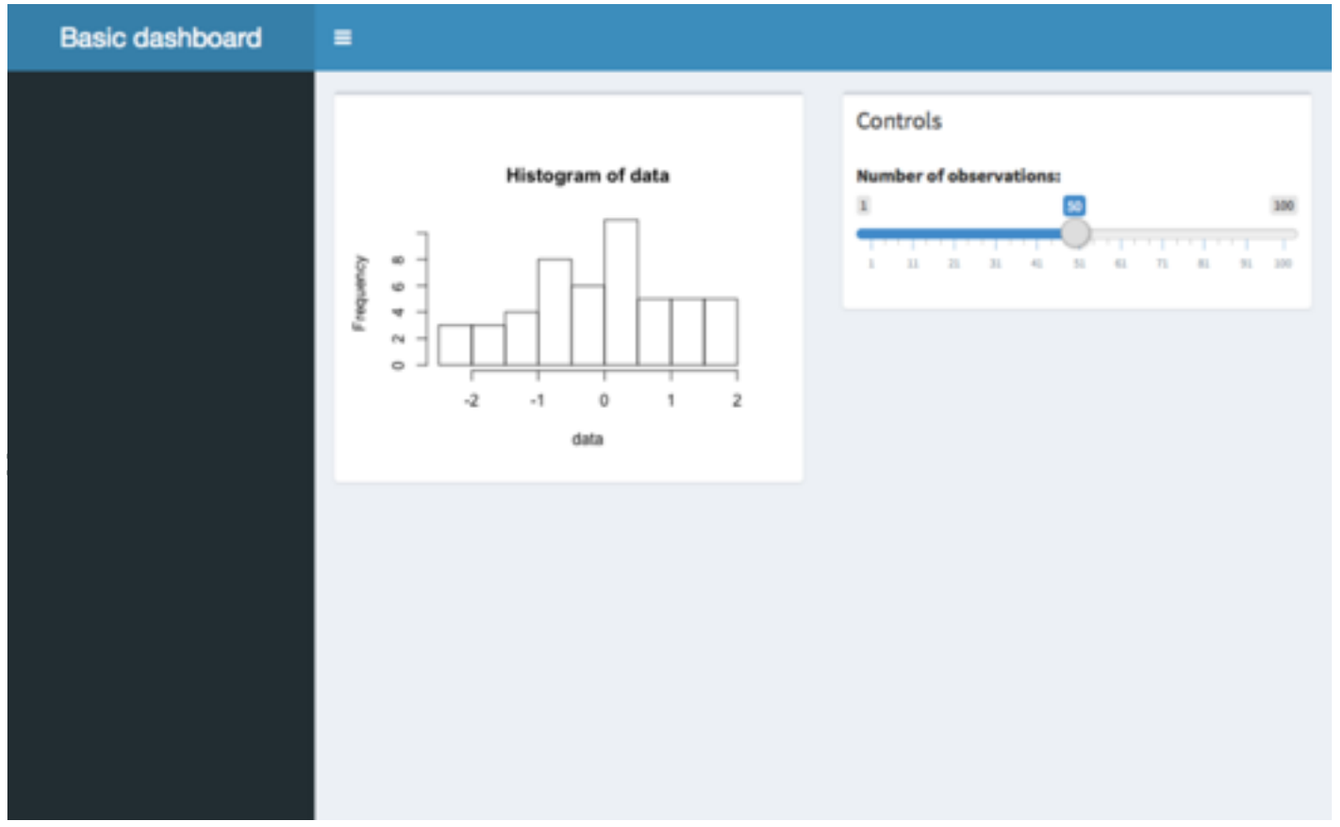
```
# Install devtools if needed
# install.packages("devtools")

devtools::install_github("rstudio/shinydashboard")
```

Documentation at:
http://rstudio.github.io/shinydashboard/

```r
## app.R (single-file app)

library(shiny)
library(shinydashboard)

ui <- dashboardPage(
  dashboardHeader(),
  dashboardSidebar(),
  dashboardBody()
)

server <- function(input, output) {
}

shinyApp(ui, server)
```

```
## app.R (single-file app)

library(shiny)
library(shinydashboard)

ui <- dashboardPage(
  dashboardHeader(),
  dashboardSidebar(),
  dashboardBody()
)

server <- function(input, output) {
}

shinyApp(ui, server)
```



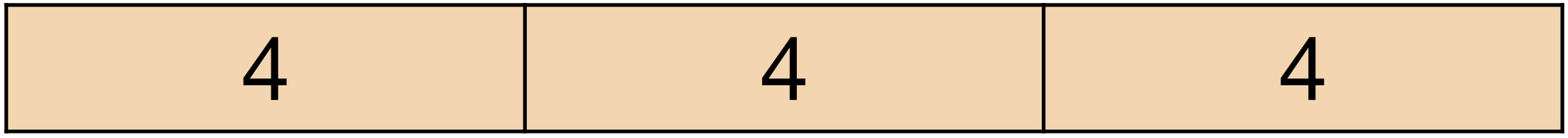



```
ui <- dashboardPage(
  dashboardHeader(title
  dashboardSidebar(),
  dashboardBody(
    fluidRow(
      box(plotOutput("plot1", height = 250)),
      box(
        title = "Controls",
        sliderInput("slider", “Observations:",
                    min=1, max=100, value=50)
      )
    )
  )
)
```

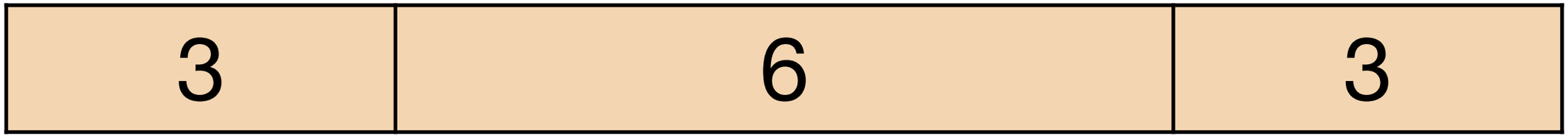
The Bootstrap grid

- Layout uses a grid of rows and columns
- Each row has 12 columns
- HTML elements can occupy any of the 12 columns

```
fluidRow(column(4, ...), column(4, ...), column(4, ...))
```

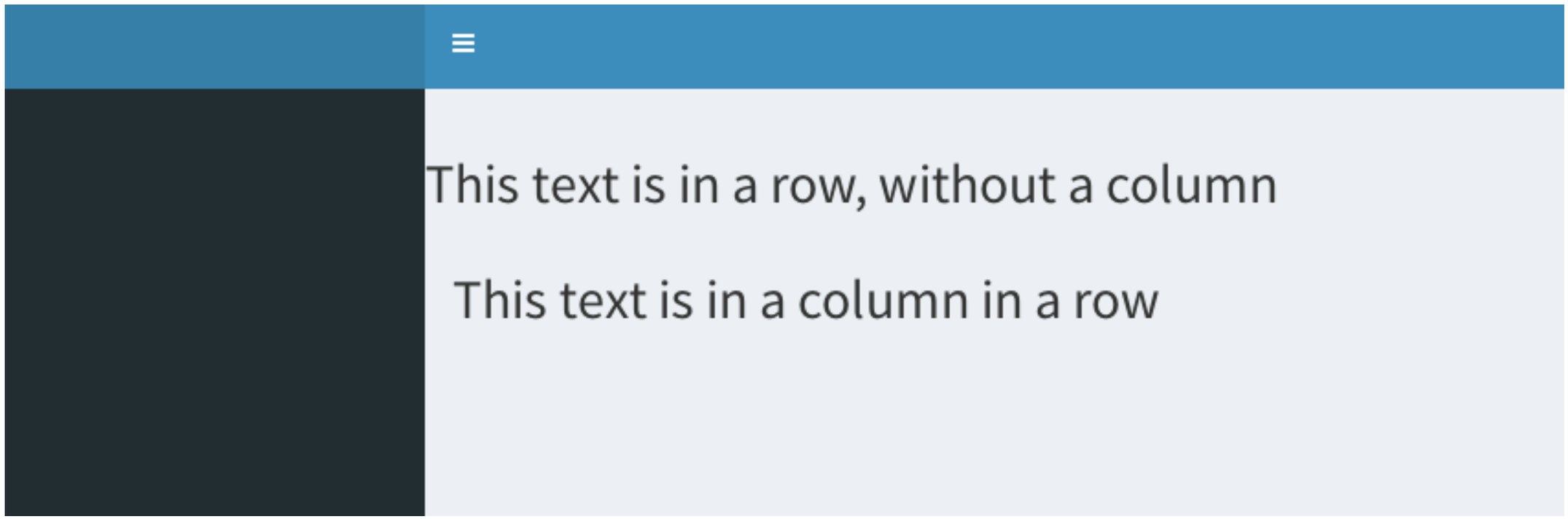


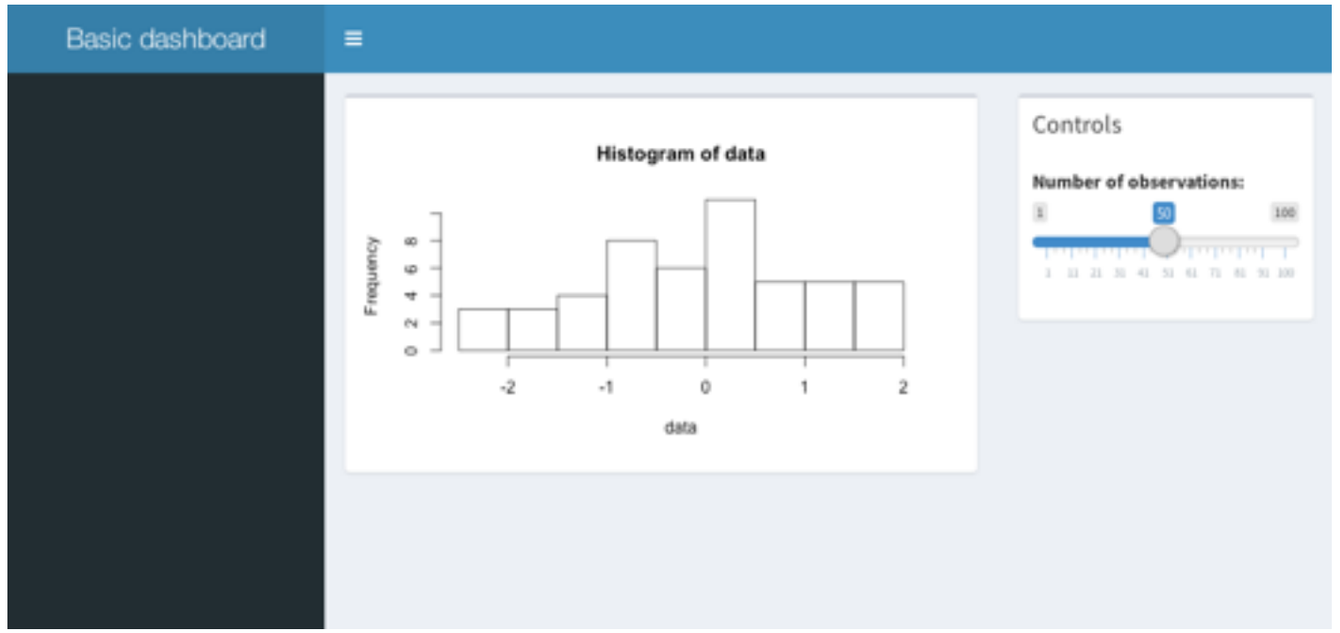

```
fluidRow(column(3, ...), column(6, ...), column(3, ...))
```




```
fluidRow(column(4, ...), column(8, ...))
```

```
# Don’t forget your columns!
dashboardBody(
  fluidRow(
    h2("This text is in a row, without a column")
  ),
  fluidRow(
    column(width = 12,
      h2("This text is in a column in a row")
    )
  )
)
```

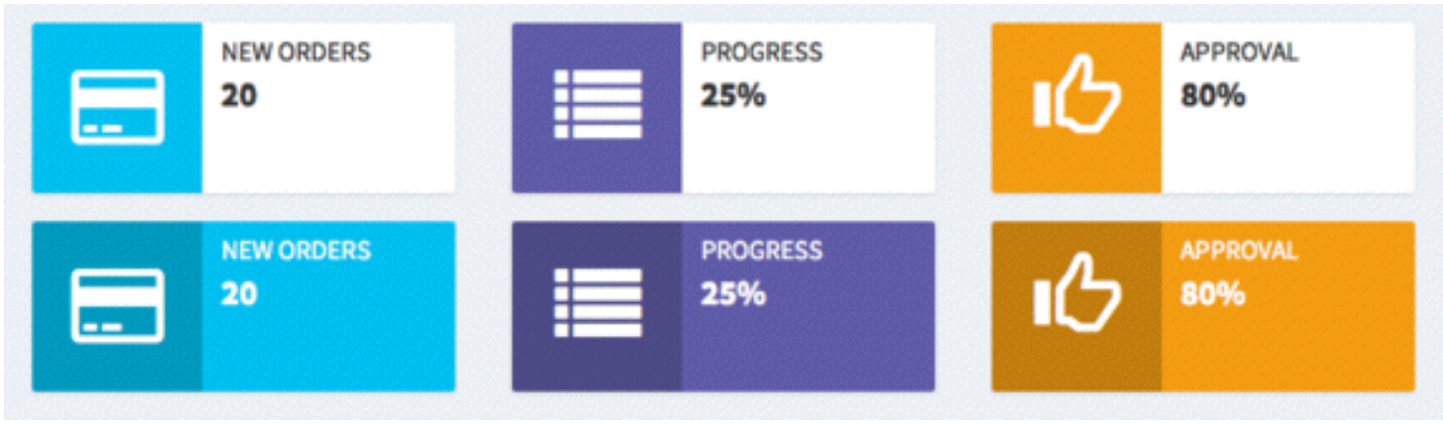




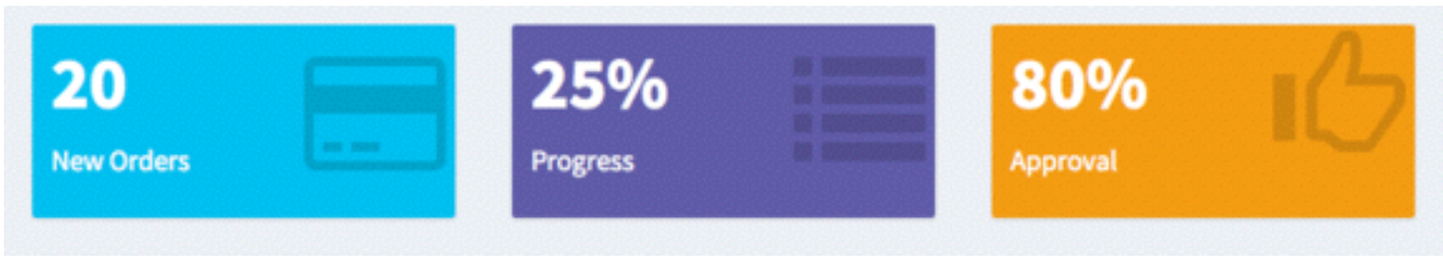

```
ui <- dashboardPage(
  dashboardHeader(title =
  dashboardSidebar(),
  dashboardBody(
    fluidRow(
      box(width = 8, plotOutput(“plot1”, height = 250)),
      box(
        width = 4,
        title = "Controls",
        sliderInput("slider", “Observations:",
                    min=1, max=100, value=50)
      )
    )
  )
)
```

Other kinds of boxes

`infoBox()`



`valueBox()`

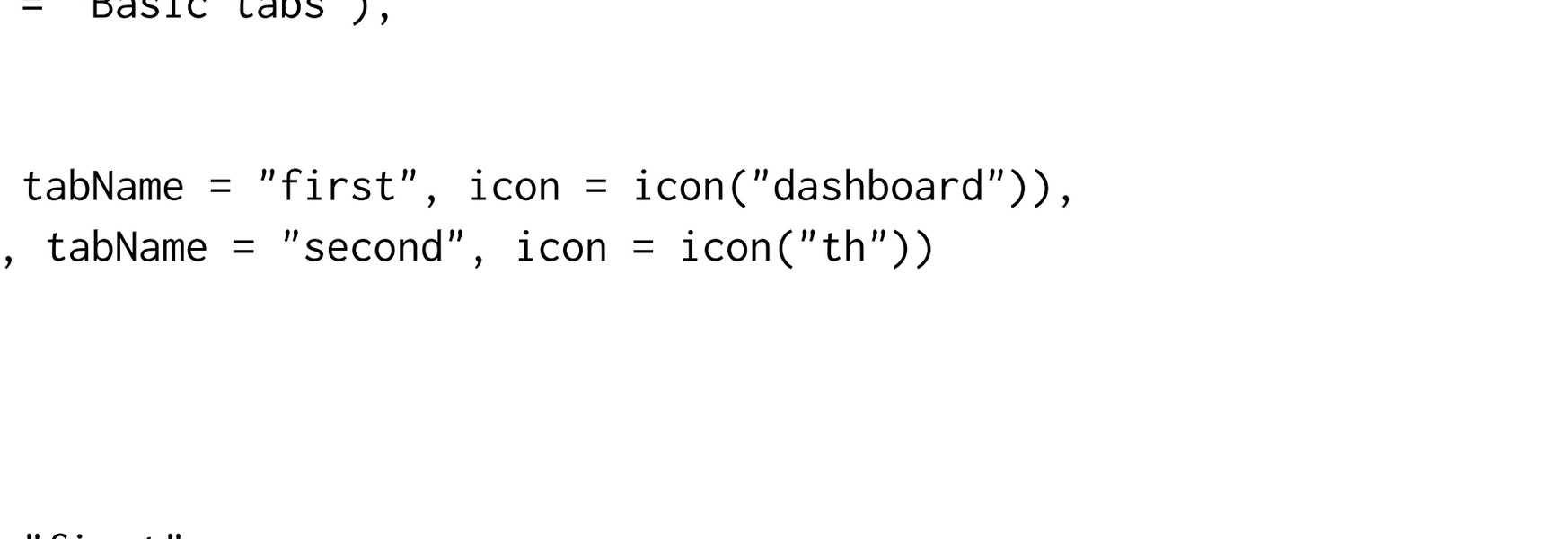


http://rstudio.github.io/shinydashboard/

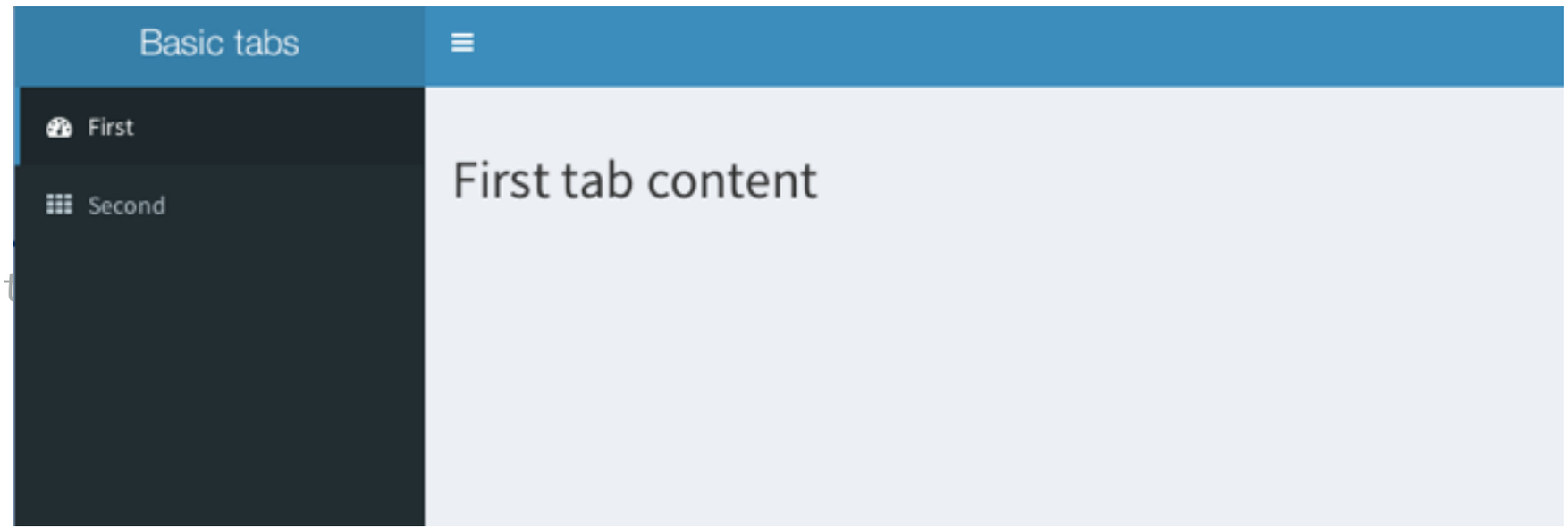
Sidebar

- Tab items
- Inputs

```
library(shiny)
library(shinydashboard)

ui <- dashboardPage(
  dashboardHeader(title = "Basic tabs"),
  dashboardSidebar(
    sidebarMenu(
      menuItem("First", tabName = "first", icon = icon("dashboard")),
      menuItem("Second", tabName = "second", icon = icon("th"))
    )
  ),
  dashboardBody(
    tabItems(
      tabItem(tabName = "first",
        h2("First tab content")
      ),
      tabItem(tabName = "second",
        h2("Second tab content")
      )
    )
  )
)

server <- function(input
shinyApp(ui, server)
```

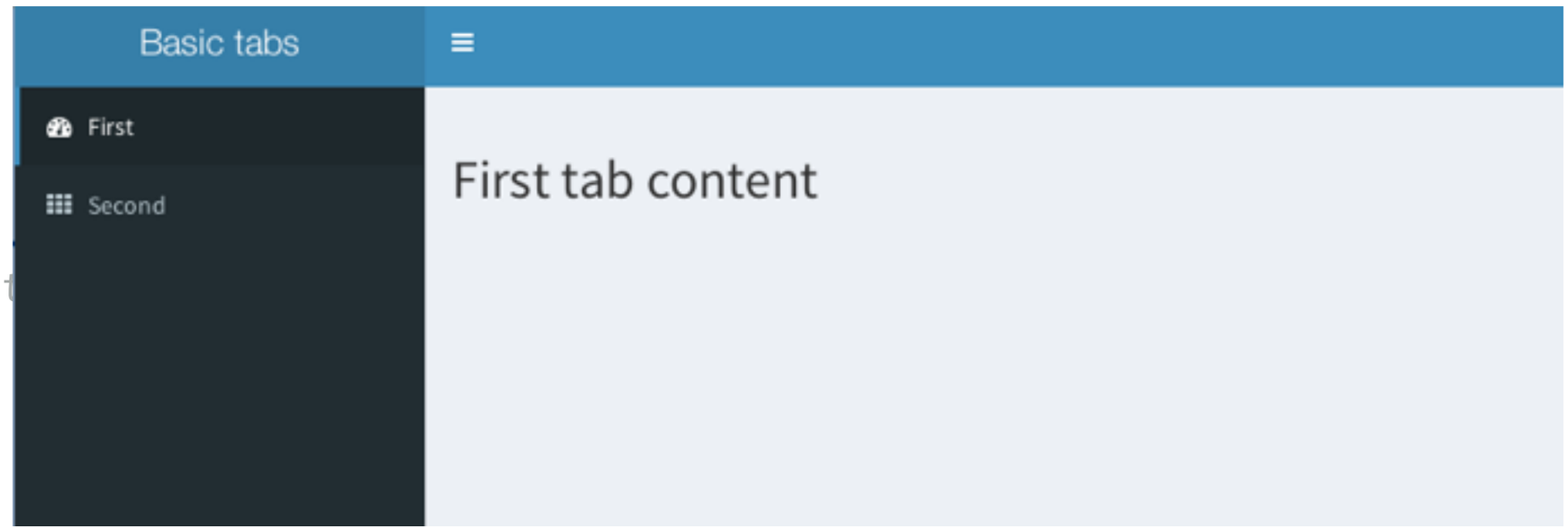


```r
library(shiny)
library(shinydashboard)

ui <- dashboardPage(
  dashboardHeader(title = "Basic tabs"),
  dashboardSidebar(
    sidebarMenu(
      menuItem("First", tabName = "first", icon = icon("dashboard")),
      menuItem("Second", tabName = "second", icon = icon("th"))
    )
  ),
  dashboardBody(
    tabItems(
      tabItem(tabName = "first",
        h2("First tab content")
      ),
      tabItem(tabName = "second",
        h2("Second tab content")
      )
    )
  )
)

server <- function(input
shinyApp(ui, server)
```



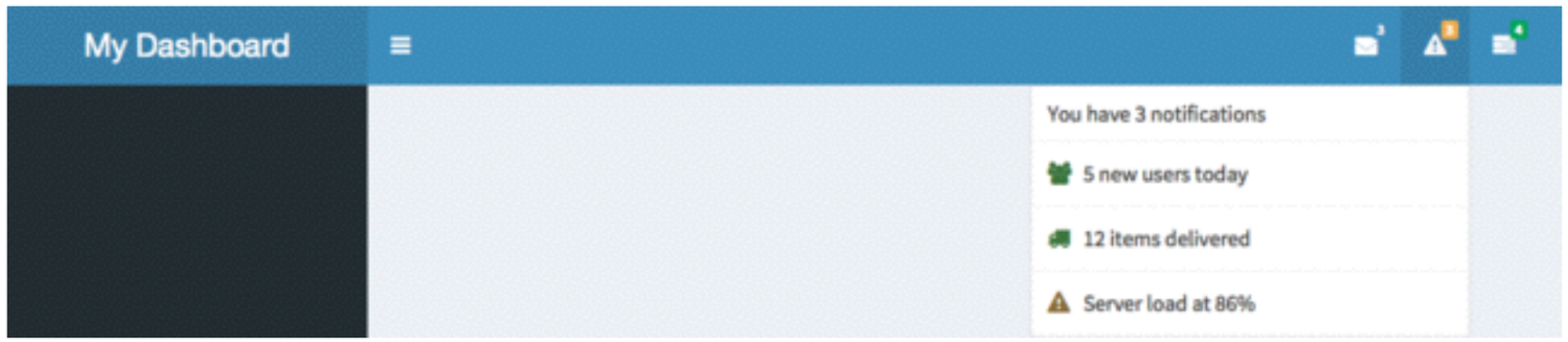
```
library(shiny)
library(shinydashboard)

ui <- dashboardPage(
  dashboardHeader(title = "Basic tabs"),
  dashboardSidebar(
    sidebarMenu(
      menuItem("First", tabName = "first", icon = icon("dashboard")),
      menuItem("Second", tabName = "second", icon = icon("th"))
    )
  ),
  dashboardBody(
    tabItems(
      tabItem(tabName = "first",
        h2("First tab content")
      ),
      tabItem(tabName = "second",
        h2("Second tab content")
      )
    )
  )
)

server <- function(input
shinyApp(ui, server)
```

Icons are from Font Awesome. See:
http://rstudio.github.io/shinydashboard/appearance.html



Header

- Title
- Message/notification/task menus



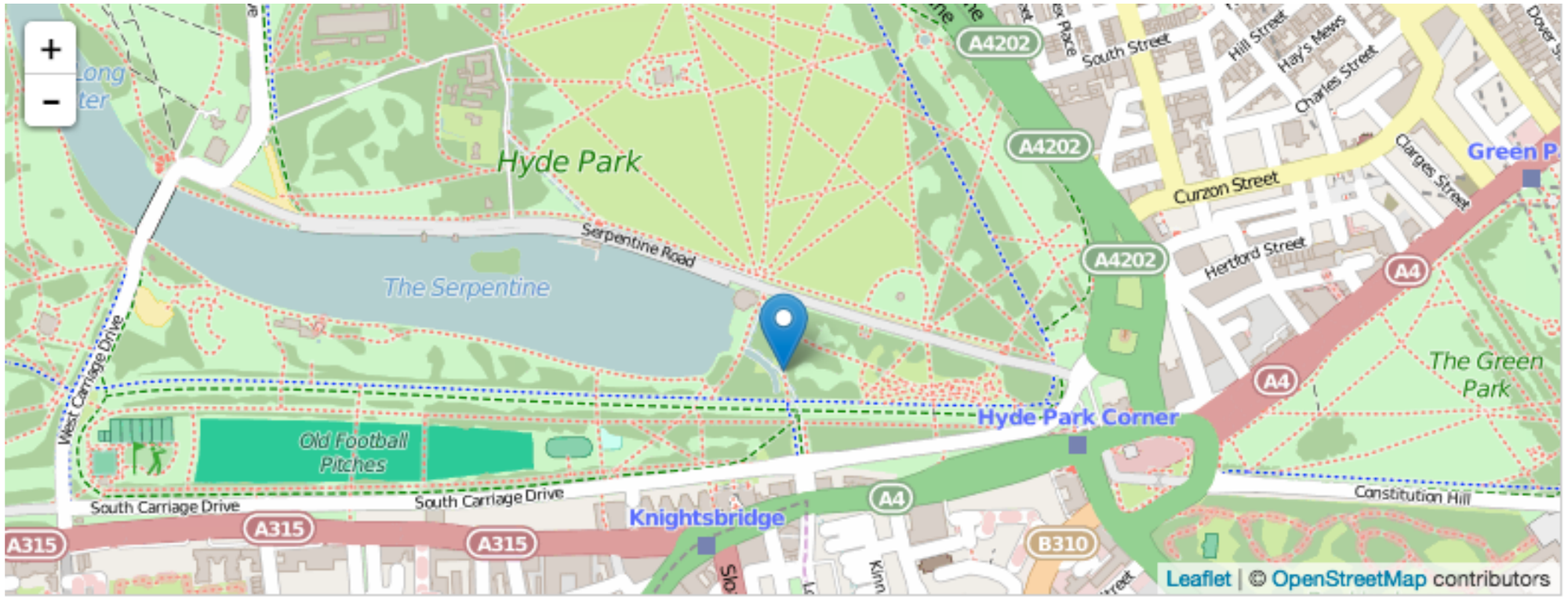
Leaflet

Maps in R

- Good tools for working with map data
- Map output is decent
- Hard to interact with

Maps in the browser

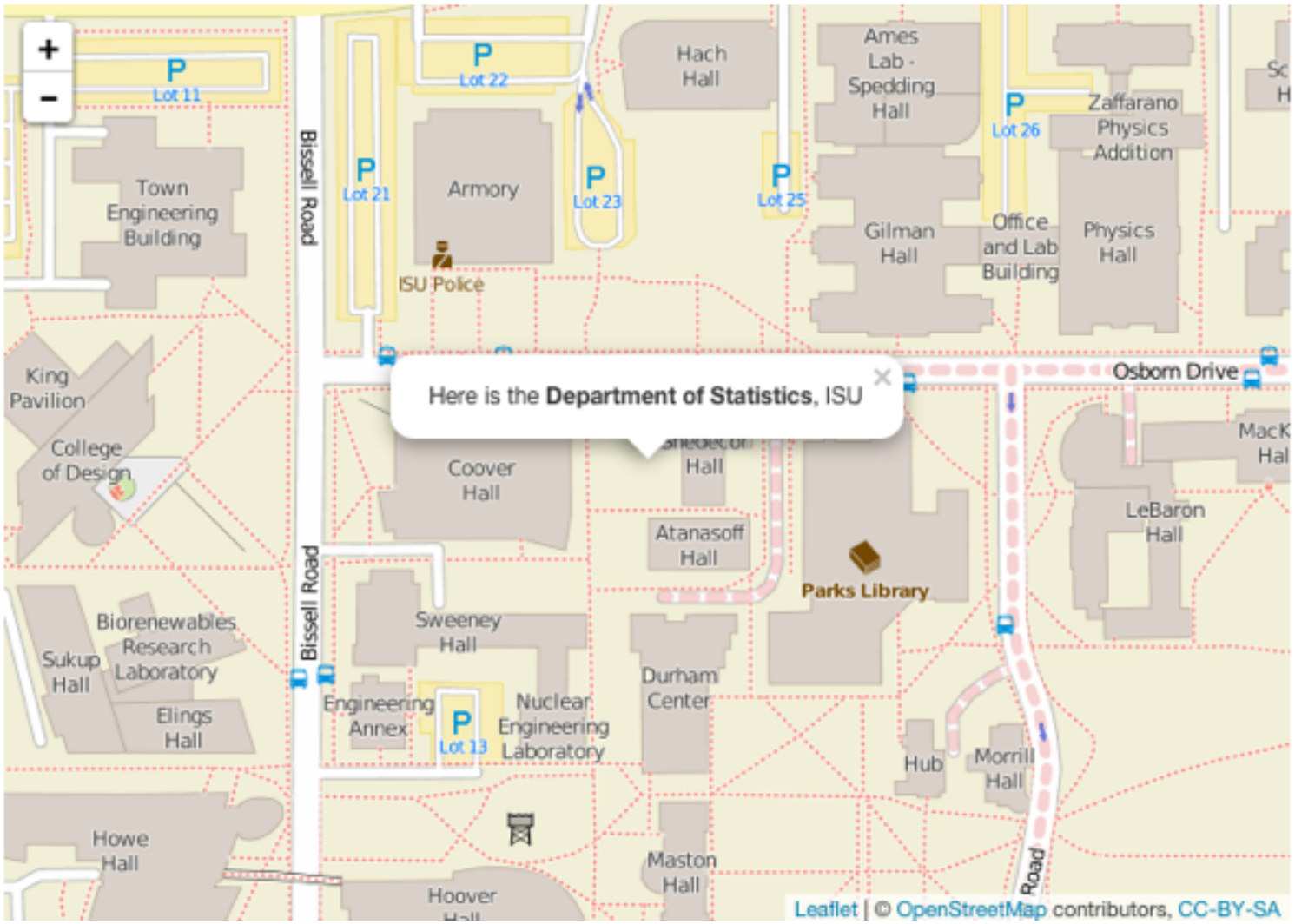
- leafletjs: Javascript library for interactive maps
- http://leafletjs.com/



leaflet

- R package that provides a nice interface to leafletjs
- http://rstudio.github.io/leaflet/

```
library(leaflet)
leaflet() %>%
  addTiles() %>%
  setView(-93.65, 42.0285, zoom = 17) %>%
  addPopups(-93.65, 42.0285, 'Here is the
<b>Department of Statistics</b>, ISU')
```



shinydashboard

test.R | Untitled1

Source on Save | Run | Source

```
1
```

1:1 (Top Level) R Script

Environment | History | Build | Git

Import Dataset | List

Global Environment

Environment is empty

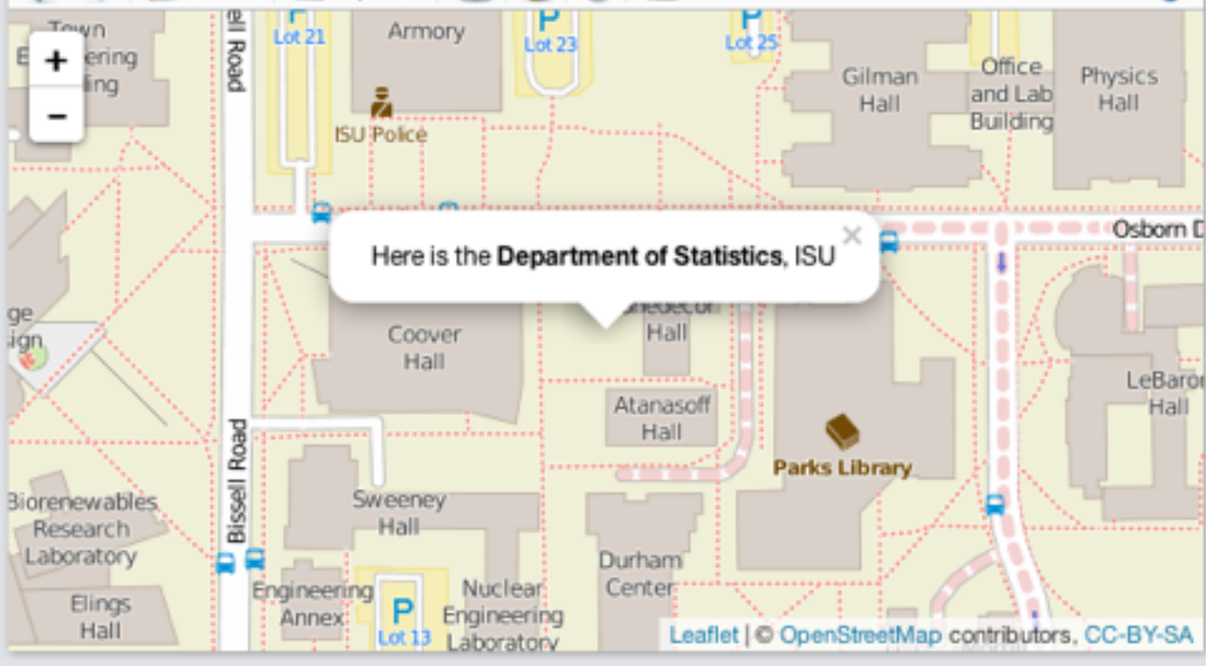
Console | Find in Files

~/Dropbox/Projects/shinydashboard/

```
> library(leaflet)
> leaflet() %>%
+   addTiles() %>%
+   setView(-93.65, 42.0285, zoom = 17) %>%
+   addPopups(-93.65, 42.0285, 'Here is the <b>Department of Statistics</b>, ISU'
)
>
>
```

Files | Plots | Packages | Help | Viewer

Zoom | Export



Leaflet in a Shiny app

```
## UI code ##
leafletOutput("map")
```

```
## Server code ##
output$map <- renderLeaflet({
  leaflet() %>%
    addTiles() %>%
    setView(-93.65, 42.0285, zoom = 17)
})
```

Demo dashboard

- Activity dashboard
- Data from www.pilrhealth.com

Deploying your dashboard

Deployment options

- Shiny Server (open source)
- Shiny Server Pro
- shinyapps.io

Run on your own Linux server

Hosted by RStudio in the cloud

Shiny Server Open Source

- Free software (AGPL v3)
- Run on your own Linux server
- No authentication or SSL
- One R process per app

Shiny Server Pro

- Commercial license
- Run on your own Linux server
- Authentication and SSL
- Multiple R processes per app
- Admin/monitoring dashboard

shinyapps.io

- Hosted by RStudio in Amazon AWS
- Supports SSL and authentication
- Multiple R processes per app
- Admin/monitoring dashboard
- No persistent storage yet

Deploying to shinyapps.io

```
devtools::install_github("rstudio/shinyapps")
```

Next, create an account at shinyapps.io, and configure the shinyapps package.

See the Getting Started Guide:

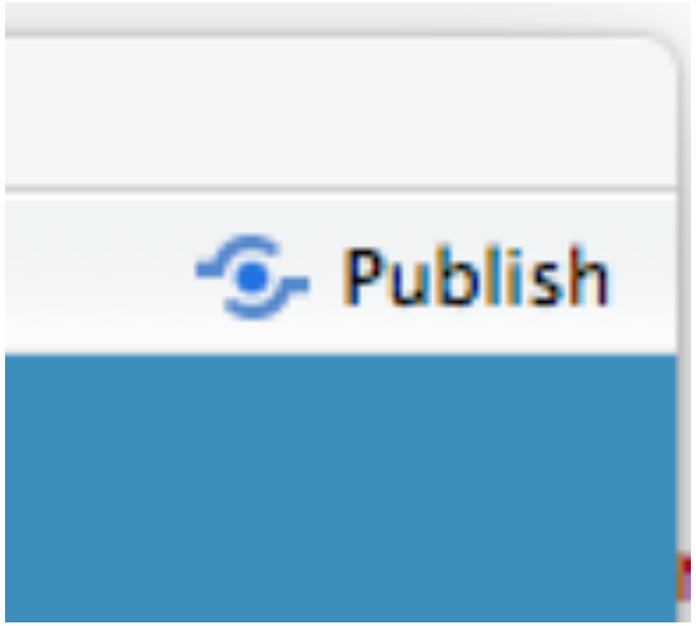
http://shiny.rstudio.com/articles/shinyapps.html

```
shinyapps::deployApp("appdir")
```

The server will use the same version of R and install all the same versions of packages.

Deploying to shinyapps.io

Another alternative: click the Publish button



(Might require RStudio daily preview build)

Resources

- Dashboard demo: https://winston.shinyapps.io/activity-dashboard/
- Shiny: http://shiny.rstudio.com/
- Shinydashboard: http://rstudio.github.io/shinydashboard/
- Leaflet: http://rstudio.github.io/leaflet/
- Shiny Server: http://www.rstudio.com/products/shiny/shiny-server/
- shinyapps.io: http://www.shinyapps.io/
- AdminLTE: https://almsaeedstudio.com/themes/AdminLTE/index2.html