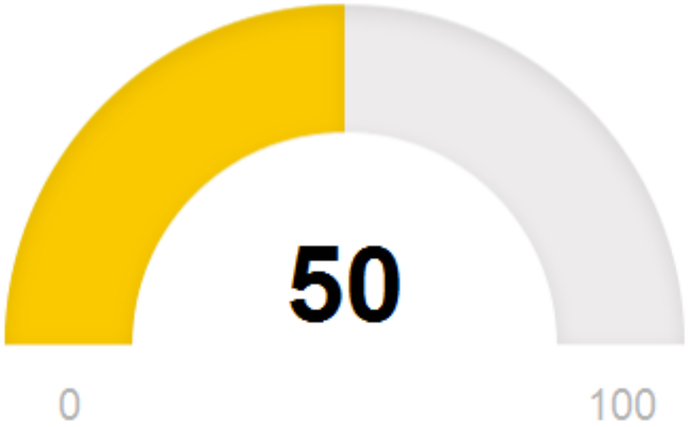
250 Northern Ave, Boston, MA 02210
Phone: 844-448-1212
Email: info@rstudio.com
Web: http://www.rstudio.com

Building Dashboards

Using reactiveFileReader, shinydashboard, and htmlTemplate



Nathan Stephens
Director of Solutions Engineering
January 2016
Email: nathan@rstudio.com

What is a dashboard?

A dashboard is an app that:

- Is **always accessible**
- Displays **key information**
  (e.g. statistics, insights, data summaries, etc.)
- Typically **refreshes automatically** or on a schedule
- May or may not be **interactive**

Dashboards are ubiquitous!

Topics

1. How to automatically update your dashboard with new data

    reactiveFileReader function

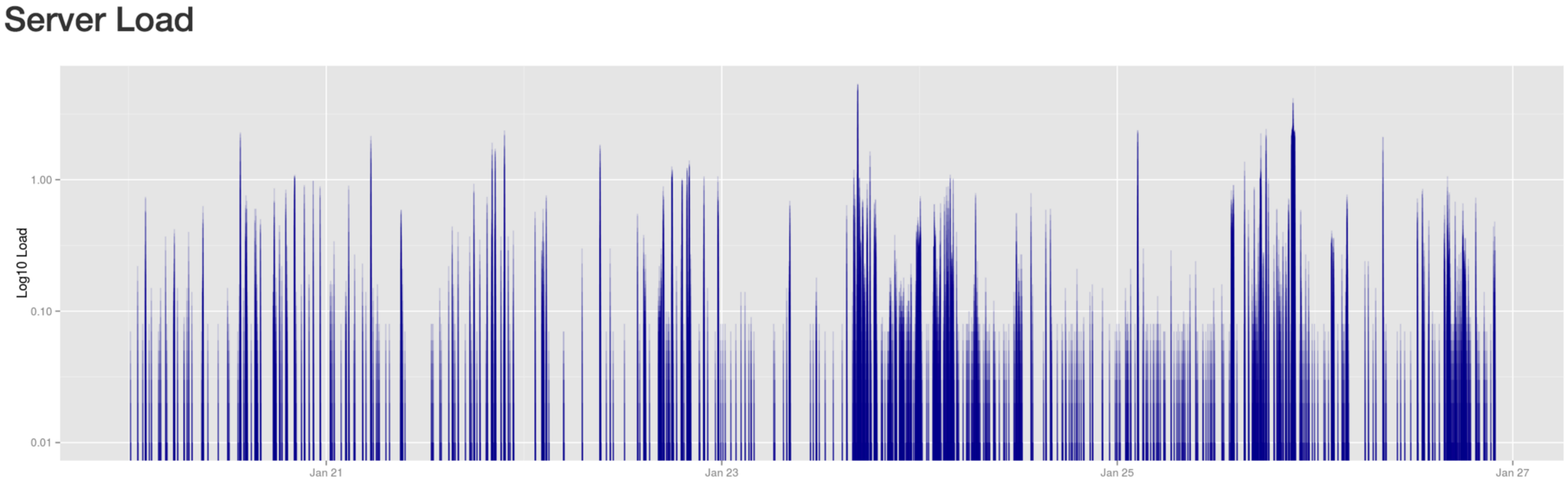
2. How to build a great dashboard UI

    shinydashboard package

htmlTemplate function

Example Use Case

Build an app that monitors server load



Create a “live” data source

serverLoad.sh

```
#!/bin/bash

MYDIR=/home/nathan/ShinyApps/serverLoad/data
FILENAME=serverLoad.txt
TABLENAME=serverLoad

LOAD=`uptime | sed 's/.*load average: //' | awk -F\, '{print $1}'`
DATE=`date +%Y-%m-%d:%H:%M:%S`
echo "$DATE,$LOAD" >> $MYDIR/$FILENAME
```

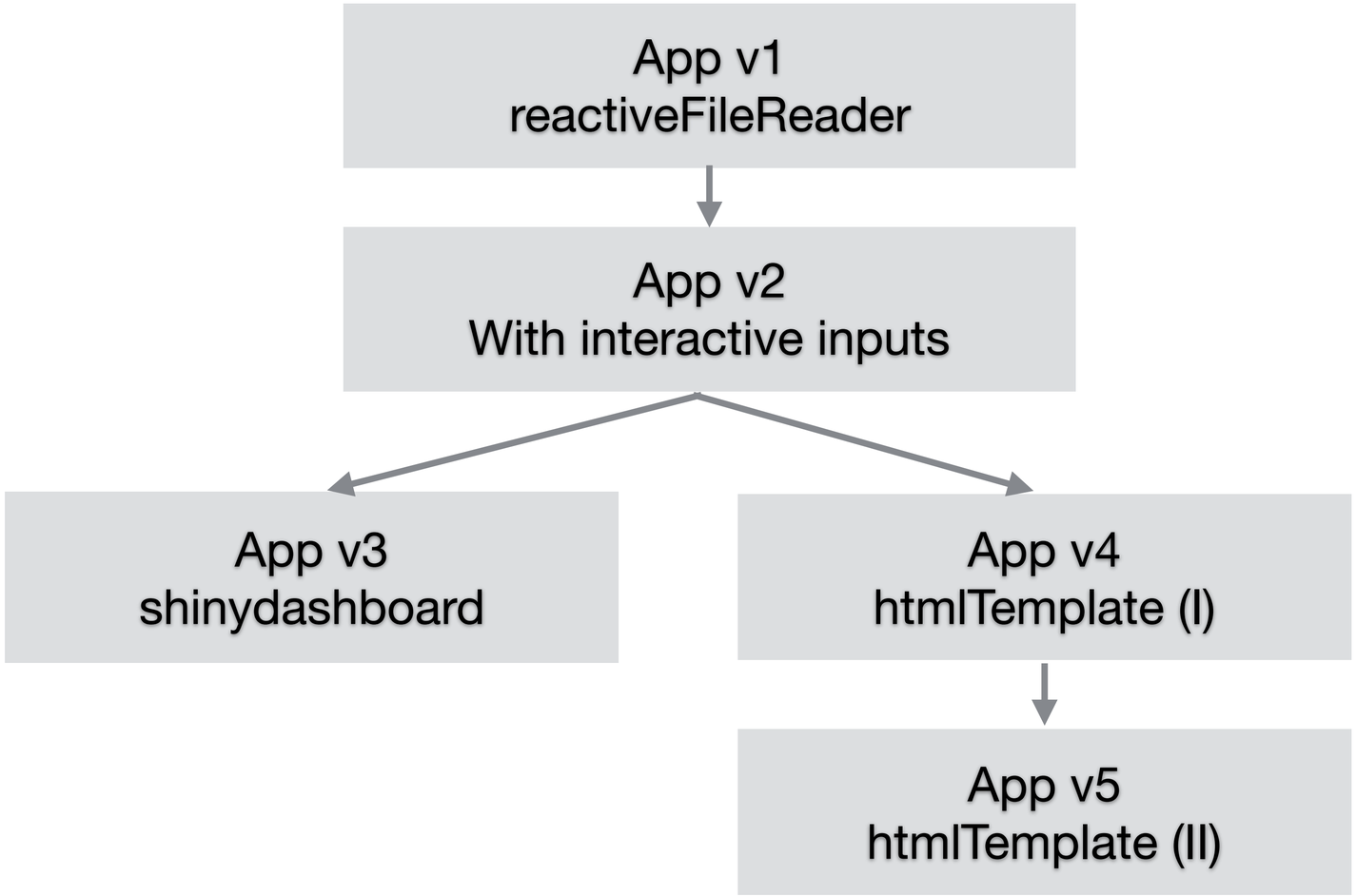
Automate data source

crontab

```
MYDIR=/home/nathan/ShinyApps/serverLoad/data
MYFILE=severLoad.sh
MYLOG=serverLoad.log

* * * * * $MYDIR/$MYFILE >> $MYDIR/$MYLOG 2>&1
* * * * * sleep 15; $MYDIR/$MYFILE >> $MYDIR/$MYLOG 2>&1
* * * * * sleep 30; $MYDIR/$MYFILE >> $MYDIR/$MYLOG 2>&1
* * * * * sleep 45; $MYDIR/$MYFILE >> $MYDIR/$MYLOG 2>&1
```

5 Versions of Server Load App



Getting Started

reactiveFileReader

```
fileReaderData <- reactiveFileReader(
  intervalMillis = 500,
  session = session,
  filePath = infile,
  readFunc = read_csv,
  col_names = c('dte', 'Load')
)
```

Server Load App
Version 1

reactiveFileReader

Server Load App
Version 2

with interactive inputs

shinyDashboard

https://rstudio.github.io/shinydashboard/

- An R package designed specifically to help you create dashboards with Shiny
- Based on bootstrap
- Has a specific structure
- Some elements can be customized (e.g skins, CSS, colors, etc.)

Format the UI with shinydashboard

shinyUI

```
ui <- shinyUI(

  fluidPage(

    titlePanel("Server Load"),

    tags$hr(),

    dateRangeInput(
      inputId = "dateRange",
      label = "Select a date range",
      start = Sys.Date(),
      end = Sys.Date(),
      separator = 'through'
    ),

    checkboxInput("ma", "Fit a moving average", FALSE),

    conditionalPanel(
      condition = "input.ma == true",
      sliderInput(
        inputId = "ma_adjust",
        label = "Moving average adjustment",
        min = 1, max = 100, value = 15, step = 1.0,ticks = FALSE
      )
    ),

    plotOutput("plot"),

    dataTableOutput("data")

  )

)
```

dashboardPage

```
ui <- dashboardPage(
  skin = 'blue',
  dashboardHeader(title = "Server Load"),
  dashboardSidebar(
    sidebarMenu(
      menuItem("Calendar", tabName = "settings", icon = icon("calendar")),
      menuItem("Plot", tabName = "plot", icon = icon("image")),
      menuItem("Data", tabName = "data", icon = icon("table"))
    )
  ),
  dashboardBody(
    tabItems(
      tabItem(
        tabName = "settings",
        valueBoxOutput("valueBoxDate"),
        valueBoxOutput("valueBoxValue"),
        valueBoxOutput("valueBoxRecs"),
        dateRangeInput(
          "dateRange", "Select a date range",
          Sys.Date(), Sys.Date(), separator = 'through')
      ),
      tabItem(
        tabName = "plot",
        fluidRow(
          box(
            'Time Series', width = 8, status = 'primary',
            plotOutput("plot", width = "100%")
            ),
          box(
            'Moving Average', width = 4, status='info',
            checkboxInput("ma", "Fit a moving average", FALSE),
            conditionalPanel(
              condition = "input.ma == true",
              sliderInput(
                inputId = "ma_adjust",
                label = "Moving average adjustment",
                 min = 1, max = 100, value = 15, step = 1.0)
            )
          )
        )
      ),
      tabItem(
        tabName = "data",
        fluidRow(
          box(
            'Raw Data', width = 8, status='primary', dataTableOutput("data")
            )
          )
        )
      )
    )
  )
```

Server Load App
Version 3

shinydashboard

htmlTemplate Recap

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

?

Format the UI with htmlWidgets

shinyUI

```
ui <- shinyUI(
  fluidPage(
    …
    plotOutput("plot"),
    dataTableOutput("data")
  )
)
```

htmlTemplate

```
ui <- htmlTemplate(


  …
  plot = plotOutput("plot"),
  data = dataTableOutput("data")
  )
```

Insert the Shiny Elements Into HTML

Complete web pages

To use an HTML template for the UI, first create an HTML file in your app directory, at the same level as the `ui.R`, `server.R`, or `app.R` files (not in a `www/` subdirectory). Here's an example template for a complete web page, `template.html`:

```
<!DOCTYPE html>
<!-- template.html -->
<html>
  <head>
    {{ headContent() }}
  </head>
  <body>
    <div>
      {{ button }}
      {{ slider }}
   </div>
  </body>
</html>
```

And here's a corresponding `ui.R` that uses the template:

```
## ui.R ##
htmlTemplate("template.html",
  button = actionButton("action", "Action"),
  slider = sliderInput("x", "X", 1, 100, 50)
)
```

Server Load App
Version 4

Education htmlTemplate

Server Load App
Version 5

Finance htmlTemplate

Conclusion

Use **reactiveFileReader** functions to automatically update your app with new data.

Use the **shinydashboard** package to format your UI with a specific structure.

Use the **htmlTemplate** function to embed your shiny dashboard elements into your organization's HTML template.