FE8828 Programming Web Applications in Finance

Week 5: 10. Shiny/3: Advanced 11. Building Financial Applications

Dr. Yang Ye yy@runchee.com

Nanyang Business School

Oct 15, 2020

Section 1

Lecture 10: Shiny/3: Advanced

Review:

- `ui`: Run once per session.
- `server`: Run once per session.
- Session: simply, new session with each browser tab open.
- Different tab, *Different session*, even in the same browser, on the same machine.
- Code inside a reactive function runs with every input change.

Review: Reactivity

- `render*`: Allow binding of one output to one/multiple inputs

```
# builds reactivity from input$data => output$hist
output$hist <- renderPlot({
  hist(input$data)
})

output$stat <- renderPlot({
  summary(input$data)
})
```

Review: observeEvent

- Allow binding of multiple outputs to one inputs.

```
actionButton(inputId = "go", label = "Click me")

observeEvent(input$go, {
  # Use of isolate to *peek* the value not to react to it.
  num_input <- input$num_input

  output$plot1 <- renderPlot({
    # if we use input$num_input here, we build a direct reactive li
    # between output$plot1 and input$num_input. This is not what we
    plot(1:number_input, runif(num_input))
  })

  output$table1 <- renderTable({ ... })
})
```


Review: reactive()

- reactiveVal(): create reactive value.
- reactive(): build reactivity.

```
# Build a reactive value a
a <- reactiveVal(0)

# This builds a reactivity from a to b.
# Once a changes, b is changed.
b <- reactive({ a() + 1 })

# Update the value of a
a(3)
# Use isolate() to extract the value
# b is updated
print(isolate(a()))
## [1] 3
print(isolate(b()))
## [1] 4
```

Review: Output with `render*()` functions

- `output$hist <- render*({...})`
  - It contains code used to create output
  - It re-runs the code with every change in the input
- Static table from df, mat, etc.s
  - renderTable()/tableOutput()
- Interactive table from data frame, matrix or other table-like structure
  - renderDataTable()/dataTableOutput()
- Plot
  - renderPlot()/plotOutput()

New topics today for Shiny

- renderText/renderPrint: print out plain running result.
- `update***Input` functions: You can also update input with new content.
- Dynamic UI to create inputs and outputs dynamically.
- Graphics output: gridExtra
- Table output: kableExtra

New render/output pair

- Get print result
  - renderText() with verbatimTextOutput() or textOutput()
- Get continous output
  - renderPrint() with verbatimTextOutput() or textOutput()
- Customized UI elements
  - uiOutput()/renderUI()

renderText()/renderPrint() example

```
# shiny-51-renderPrint.R
library(shiny)

ui <- fluidPage(
  actionButton('go', 'Go'),
  verbatimTextOutput("t1"),
  verbatimTextOutput("t2"),
)

server <- function(input, output, session) {
  observeEvent(input$go, {
    for (i in 1:10) {
      output$t1 <- renderText({ i })
    }
  })

  observeEvent(input$go, {
    output$t2 <- renderPrint({
      for (i in 1:10) {
        cat(paste0(i, "\n"))
      }
    })
  })
}
```

update***Input

- Update various input values
  - updateSelectionInput(...)
  - updateNumericInput(...)

update***Input Example

```
# shiny-36-update-min.R

library(shiny)

ui <- fluidPage(
  numericInput("shock", "Shock", value = round(runif(1) * 1000), 0),
  actionButton("add", "Add"),
  checkboxGroupInput("scenarios", "Scenarios", choices = c(), selected = c()),
  verbatimTextOutput("o1")
)

scenarios <- c(-100, -50, 0, 50, 100, 200)

server <- function(input, output, session) {
  updateCheckboxGroupInput(session, "scenarios",
                           choices = scenarios,
                           selected = scenarios)
}

shinyApp(ui, server)
```

update***Input and output

- Update various input values
  - updateSelectionInput(...)
  - updateNumericInput(...)

update***Input and output: example

```
# shiny-52-update.R

library(shiny)

ui <- fluidPage(
  numericInput("shock", "Shock", value = round(runif(1) * 1000), 0),
  actionButton("add", "Add"),
  checkboxGroupInput("scenarios", "Scenarios", choices = c(), selected = c()),

  verbatimTextOutput("o1")
)

scenarios <- c(-100, -50, 0, 50, 100)
this_env <- environment()

server <- function(input, output, session) {
  updateCheckboxGroupInput(session, "scenarios",
                           choices = scenarios,
                           selected = scenarios)

  observeEvent(input$add, {
    shock <- isolate(input$shock)
    if (!(shock %in% scenarios)) {
      # <<- will try to assign a value outside current environment
      # See help of <<-
      # scenarios <<- sort(c(scenarios, shock))
      # I would prefer to use assign to be specific for which environment
      assign("scenarios", sort(c(scenarios, shock)), envir = this_env)
      updateCheckboxGroupInput(session, "scenarios",
                               choices = scenarios,
```

renderUI/uiOutput

- Dynamically creation of UI (user interface) with input and outputs.
  - Append new items to `tagList()`

Create dynamic output `tagList()`

```
# shiny-53-renderUI.R

library(shiny)

ui <- fluidPage(
  uiOutput("p1")
)

server <- function(input, output, session) {
  output$p1 <- renderUI({
    tl <- tagList(
      h1("HTML t1"),
      uiOutput("t1"),
      h1("Plot p1p1"),
      plotOutput("p1p1")
    )

    tl
  })

  output$t1 <- renderUI({
    tagList(
      h1("HTML p1t1 inside t1"),
      plotOutput("p1t1")
    )
  })

  output$p1t1 <- renderPlot({
    plot(1:100, runif(100))
  })
```

Create dynamic output 2

You can use newly created UI immediately

```
# shiny-54-renderUI-min.R

library(shiny)

ui <- fluidPage(
  uiOutput("p1")
)

server <- function(input, output, session) {
  output$p1 <- renderUI({
    uiopt <- tagList()
    for (i in 1:3) {
      uiopt <- tagList(
        uiopt,
        h1(paste0("HTML t", i)),
        h1(paste0("Plot p", i)),
        plotOutput(paste0("plot", i)))
    }
    uiopt
  })

  output[[paste0("plot", 1)]] <- renderPlot({ hist(rnorm(1000, 0, 1)) })
  output$plot2 <- renderPlot({ hist(runif(1000)) })
}

shinyApp(ui, server)
```


Dynamic output: output doesn't work?

```
# shiny-54-renderUI-output-not-working.R

library(shiny)

ui <- fluidPage(
  uiOutput("p1")
)

server <- function(input, output, session) {
  output$p1 <- renderUI({
    uiopt <- tagList()
    for (i in 1:3) {
      uiopt <- tagList(
        uiopt,
        h1(paste0("HTML t", i)),
        h1(paste0("Plot p", i)),
        plotOutput(paste0("plot", i)))
    }
    uiopt
  })

  for (i in 1:3) {
    # The code in render*() function are delayed run.
    # When it runs, i has already become 3.
    output[[paste0("plot", i)]] <- renderPlot({
      hist(rnorm(1000, 0, i * 100))
    })
  }
}
```

Dynamic output: fixed

```
# shiny-54-renderUI-output-improved.R

library(shiny)

ui <- fluidPage(
  uiOutput("p1")
)

server <- function(input, output, session) {
  output$p1 <- renderUI({
    uiopt <- tagList()

    for (i in 1:3) {
      uiopt <- tagList(
        uiopt,
        h1(paste0("HTML t", i)),
        h1(paste0("Plot p", i)),
        plotOutput(paste0("plot", i)))
    }

    uiopt
  })

  sds <- 1:3
  sds_i <- 1
  server_env <- environment()

  for (i in 1:3) {
    # The code in render*() function are delayed run.
    # When it runs, i has already become 3.
```


Dynamic output: fixed and improved

```
# shiny-54-renderUI-output-improved.R

library(shiny)

ui <- fluidPage(
  uiOutput("p1")
)

server <- function(input, output, session) {
  output$p1 <- renderUI({
    uiopt <- tagList()
    for (i in 1:3) {
      uiopt <- tagList(
        uiopt,
        h1(paste0("HTML t", i)),
        h1(paste0("Plot p", i)),
        plotOutput(paste0("plot", i)))
    }
    uiopt
  })

  sds <- 1:3
  sds_i <- 1
  server_env <- environment()

  # place the function outside
  plot_one <- function() {
    sds_i <- get('sds_i', envir = server_env)
    hist(rnorm(1000, 0, sds[sds_i] * 100))
    assign('sds_i',sds_i + 1, envir = server_env)
```

Very dynamic

```
# shiny-37-createDynamic.R

library(shiny)

ui <- fluidPage(
  uiOutput("p1"),
  verbatimTextOutput("o1")
)

server <- function(input, output, session) {
  baseList <- tagList(
    numericInput("shock", "Shock", value = round(runif(1) * 1000), 0),
    actionButton("add", "Add")
  )

  scenarios <- c(-100, -50, 0, 50, 100)
  tagl <- NA
  this_env <- environment()
  gen_ui <- function(scenarios, values = NA) {
    output$p1 <- renderUI({
      assign('tagl',baseList,envir = this_env) # Use t1 here so we can
      for (ss in 1:length(scenarios)) {
        nm <- paste0("scenarios_", ss)
        if (is.na(values[ss])) {
          val <- TRUE
        } else {
          val <- values[ss]
        }
        # we are creating a list of checkboxInput in ui
        tagl <- tagList(tagl, checkboxInput(nm, scenarios[ss], value = val))
```


ggplot/gridExtra

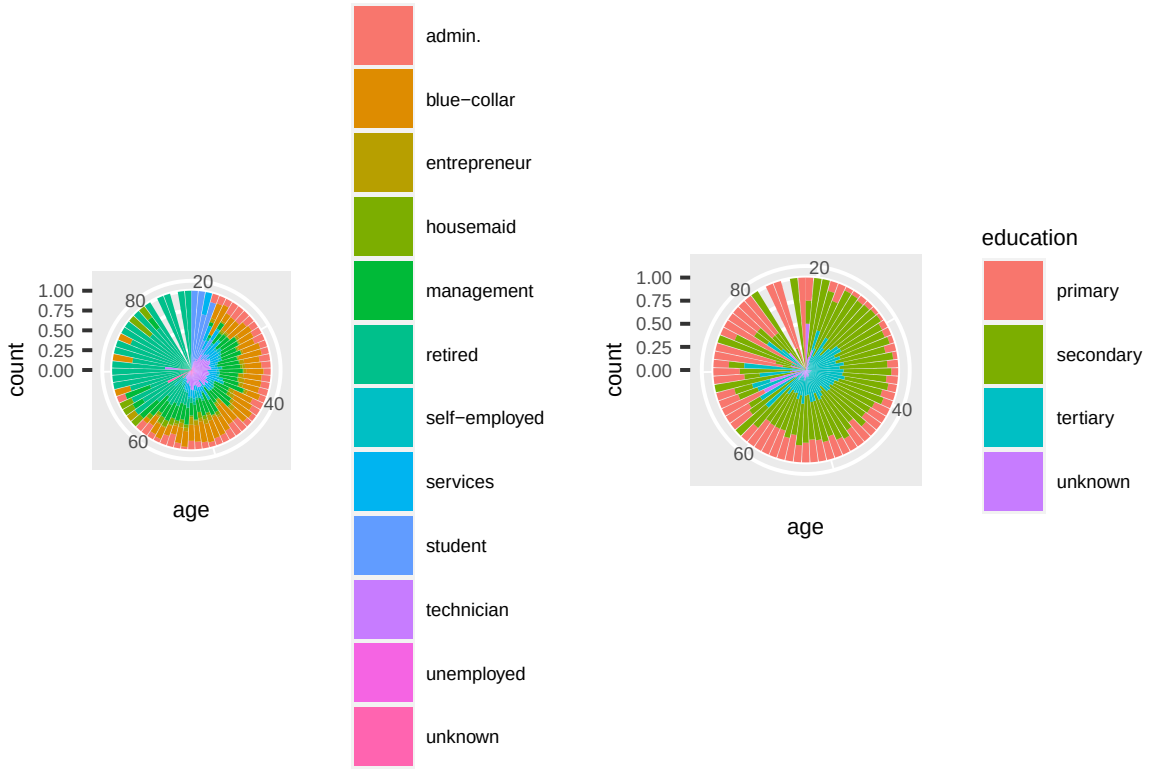
If we need to generate multiple plots. ggplot has a companion package to arrange plots.

SxS: side by side

```
library(gridExtra)
g <- ggplot(bank) + coord_polar() + theme(text = element_text(size=6))
p1 <- g+geom_bar(mapping = aes(x = age, fill = job),position = "fill")
p2 <- g+geom_bar(mapping = aes(x = age, fill = education),position = "fill")
```

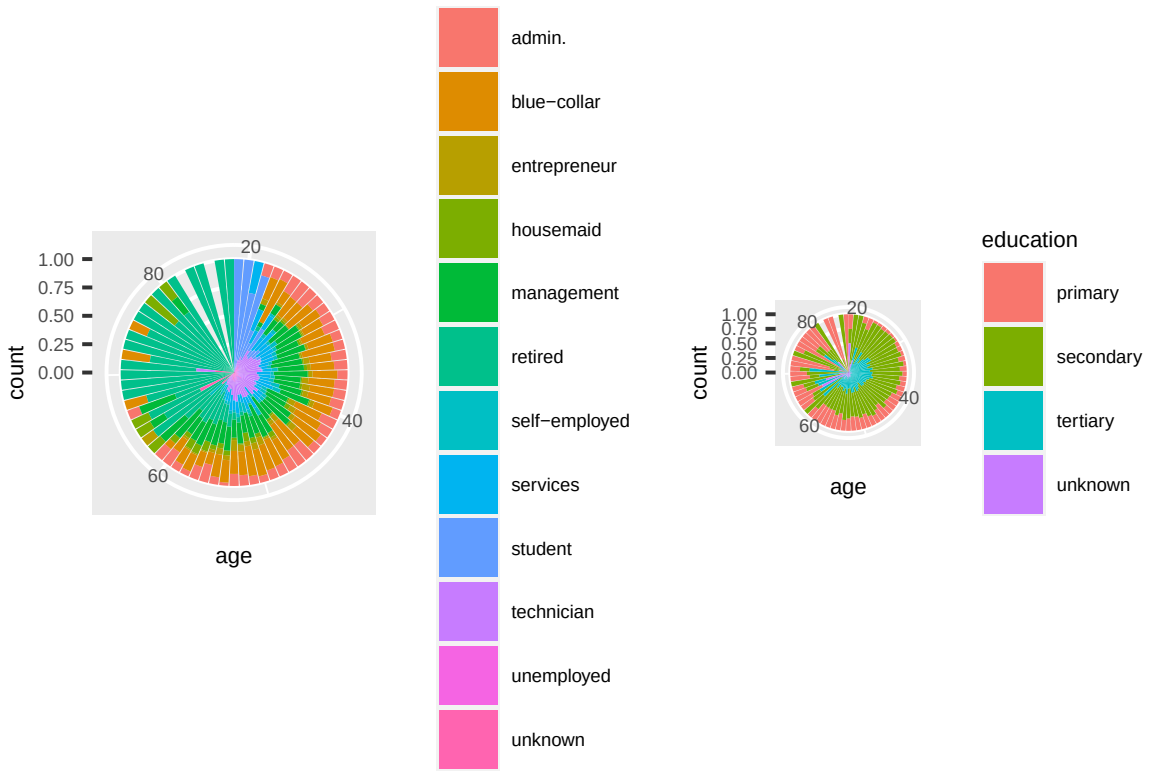
ggplot/gridExtra: example 1

```
grid.arrange(p1, p2, ncol=2, nrow=1)
```



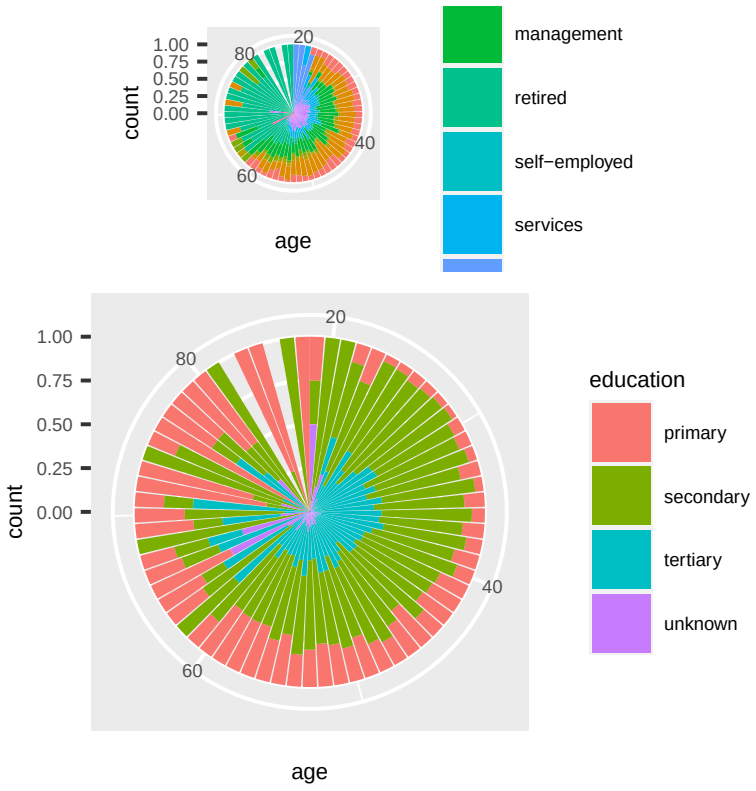
ggplot/gridExtra: example 2

```
grid.arrange(p1, p2, ncol=2, nrow=1, widths = c(4,3))
```



ggplot/gridExtra: example 3

```
grid.arrange(p1, p2, ncol=1, nrow=2, heights = c(2,4))
```



ggplot/gridExtra

```
library(tibble)
library(ggplot2)
library(gridExtra)

df <- tibble(x = rnorm(1000), y = rnorm(1000))

hist_top <- ggplot(df, aes(x = x)) + geom_density()

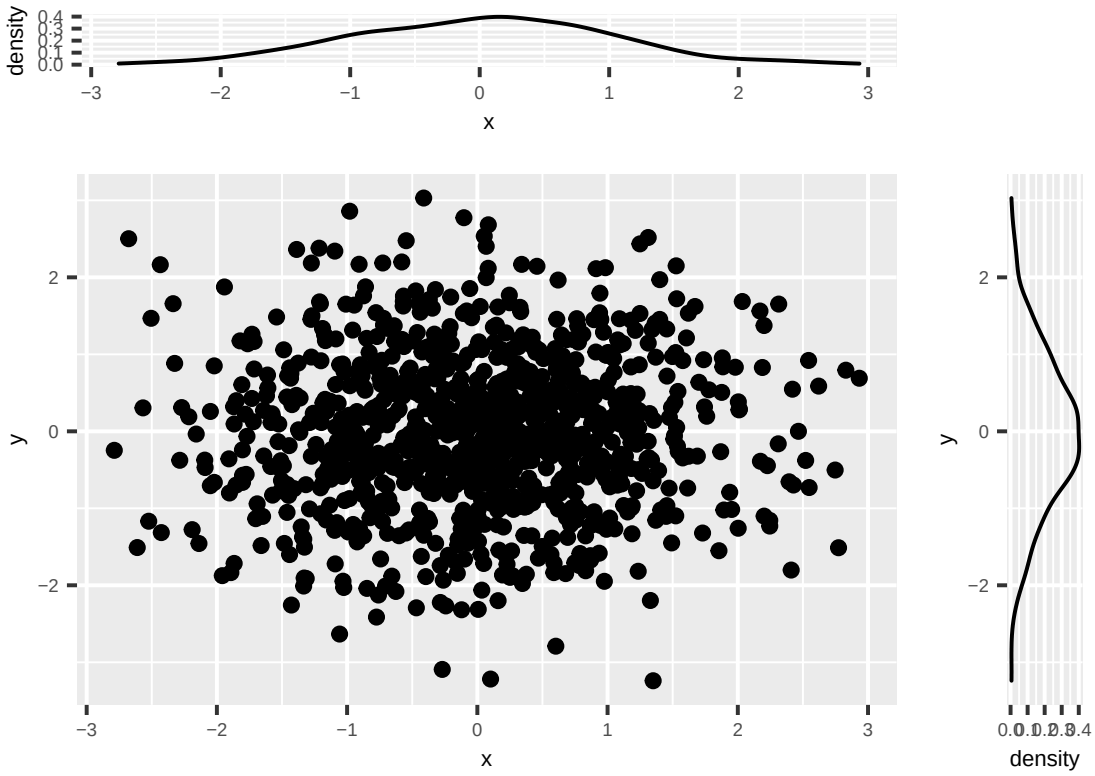
empty <-
  ggplot()+geom_point(aes(1,1), colour="white")+
  theme(axis.ticks=element_blank(),
        panel.background=element_blank(),
        axis.text.x=element_blank(), axis.text.y=element_blank(),
        axis.title.x=element_blank(), axis.title.y=element_blank())

scatter <- ggplot(df, aes(x = x, y = y)) + geom_point()

hist_right <- ggplot(df, aes(x = y)) + geom_density() + coord_flip()

grid.arrange(hist_top, empty, scatter, hist_right,
             ncol=2, nrow=2,
             widths=c(3.5, 0.7), heights=c(1, 4))
```

ggplot/gridExtra: result



knitr/kableExtra

kable is provided by `knitr` package. `kableExtra` enhance it with more functions. So we load both packages.

```
```{r shiny_block}
library(knitr)
library(kableExtra)

# This is HTML output
kable(df, format = "html")

# Use function() { } to output html
output$p1 <- function() {
  kable(df, format = "html")
}
```
```

kable_styling

- Get all styles from here https://cran.r-project.org/web/packages/kableExtra/vignettes/awesome_table_in_html.html
- style

```
mtcars[1:10, , drop = FALSE] %>%
  kbl() %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"),
                font_size = 12,
                full_width = F, # True for left-to-right width
                position = "left") # if full_width == F
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4

kable_styling: column_spec

```
mtcars[1:10, , drop = FALSE] %>%
  kbl() %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"),
                font_size = 12,
                full_width = F, # True for left-to-right width
                position = "left") %>% # if full_width == FALSE
    column_spec(1, bold = TRUE, border_right = TRUE) %>%
    column_spec(2, width = "30em", background = "yellow")
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4

kable_styling: row_spec

```
mtcars[1:10, , drop = FALSE] %>%
  kbl() %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"),
                font_size = 12,
                full_width = F, # True for left-to-right width
                position = "left") %>% # if full_width == F
    column_spec(5:7, bold = TRUE) %>%
    row_spec(3:5, bold = T, color = "white", background = "#D7261E")
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	**110**	**3.90**	**2.875**	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	**21.4**	**6**	**258.0**	**110**	**3.08**	**3.215**	**19.44**	**1**	**0**	**3**	**1**
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	**105**	**2.76**	**3.460**	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	**62**	**3.69**	**3.190**	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	**123**	**3.92**	**3.440**	18.30	1	0	4	4

Section 2

Lecture 11: Building Financial Applications

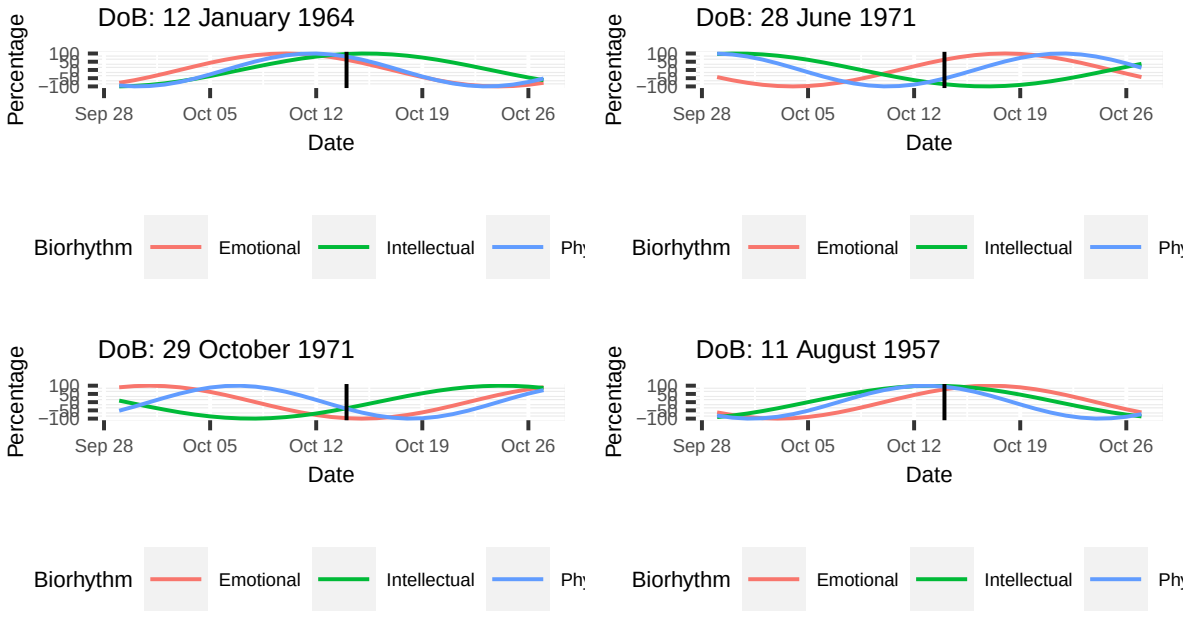
Starter

```
# biorhythm.R
suppressWarnings({library(conflicted); library(tidyverse);
conflict_prefer('lag', 'dplyr'); conflict_prefer('filter', 'dplyr')})
## [conflicted] Removing existing preference
## [conflicted] Will prefer dplyr::lag over any other package
## [conflicted] Removing existing preference
## [conflicted] Will prefer dplyr::filter over any other package

biorhythm <- function(dob, target = Sys.Date()) {
  dob <- as.Date(dob)
  target <- as.Date(target)
  t <- round(as.numeric(difftime(target, dob)))
  days <- (t - 14) : (t + 14)
  period <- tibble(Date = seq.Date(from = target - 15, by = 1, length.out = 29),
                   Physical = sin (2 * pi * days / 23) * 100,
                   Emotional = sin (2 * pi * days / 28) * 100,
                   Intellectual = sin (2 * pi * days / 33) * 100)
  period <- pivot_longer(period, cols = Physical:Intellectual, names_to = "Biorhythm", values_to
  ggplot(period, aes(x = Date, y = Percentage, col = Biorhythm)) + geom_line() +
    ggtitle(paste("DoB:", format(dob, "%d %B %Y"))) +
    geom_vline(xintercept = as.numeric(target)) +
    theme(legend.position = "bottom")
}
```


Starter - Result

```
# I took four people's birthdays. Hope they are in good mood today.
g1 <- biorhythm("1964-01-12", Sys.Date()) + theme(text = element_text(size=6))
g2 <- biorhythm("1971-06-28", Sys.Date()) + theme(text = element_text(size=6))
g3 <- biorhythm("1971-10-29", Sys.Date()) + theme(text = element_text(size=6))
g4 <- biorhythm("1957-08-11", Sys.Date()) + theme(text = element_text(size=6))
grid.arrange(g1, g2, g3, g4, ncol = 2, nrow = 2)
```



Main course

- We need following packages as a start. Use c() to install multiple packages.

```
install.packages(c("tidyquant", "Quandl","rvest",
                   "dygraphs", "forecast","testit"))
```

- `tidyquant` includes packages: `xts` (time-series data format, like data frame), `quantmod` (download prices).

tidyquant or Quandl or alphavantage?

- They can access to different data sources and different data.
- Determining factors:
  - `tidyquant/quantmod` can connect to various free services: google (unstable), yahoo (still active), mostly on stock prices.
  - `alphavantager`: stocks + FX, limit: daily 500 requests, max 5 requests per min.
  - `Quandl`: free data set is limited. Macro economic data from FRED database still available.
    - ★ US ETF/Stocks on Quandl is a premium service.
    - ★ ETF in Google/AlphaAdvantage is free.

tidyquant or Quandl?

Technical details:

- quantmod returns `xts` object. alphavantager/Quandl returns data frame or `xts`
- xts object is can `collapse` to daily, weekly, monthly price.

Tidyquant/quantmod

```
library(conflicted)
# library(tidyquant)
# conflict_prefer("filter", "dplyr")
# conflict_prefer("lag", "dplyr")

# use Google
getSymbols('SPY', src = 'yahoo', adjusted = TRUE, output.size = 'full')
## [1] "SPY"
str(SPY)
## An 'xts' object on 2007-01-03/2020-10-13 containing:
##   Data: num [1:3470, 1:6] 142 141 141 141 141 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr [1:6] "SPY.Open" "SPY.High" "SPY.Low" "SPY.Close" ...
##   Indexed by objects of class: [Date] TZ: UTC
##   xts Attributes:
## List of 2
##  $ src    : chr "yahoo"
##  $ updated: POSIXct[1:1], format: "2020-10-14 16:19:09"

# Sign up with AlphaAdvantage to get a token
# getSymbols('SPY', src = 'av', output.size = 'full', api.key = token_av)
# str(SPY)
```

Tidyquant/quantmod

```
# What's get returned?
head(SPY)
##            SPY.Open SPY.High SPY.Low SPY.Close SPY.Volume SPY.Adjusted
## 2007-01-03   142.25   142.86  140.57    141.37   94807600     106.9321
## 2007-01-04   141.23   142.05  140.61    141.67   69620600     107.1590
## 2007-01-05   141.33   141.40  140.38    140.54   76645300     106.3043
## 2007-01-08   140.82   141.41  140.25    141.19   71655000     106.7960
## 2007-01-09   141.31   141.60  140.40    141.07   75680100     106.7052
## 2007-01-10   140.58   141.57  140.30    141.54   72428000     107.0607
tail(SPY)
##            SPY.Open SPY.High SPY.Low SPY.Close SPY.Volume SPY.Adjusted
## 2020-10-06   339.91   342.17  334.38    334.93   90128900       334.93
## 2020-10-07   338.12   341.63  338.09    340.76   56999600       340.76
## 2020-10-08   342.85   343.85  341.86    343.78   45242500       343.78
## 2020-10-09   345.56   347.35  344.89    346.85   59528600       346.85
## 2020-10-12   349.59   354.02  349.06    352.43   80388500       352.43
## 2020-10-13   352.28   352.47  349.09    350.13   73163300       350.13

symbols <- c("MSFT", "AAPL")
getSymbols(symbols, src = 'yahoo', adjusted = TRUE, from = "2016-01-01")
## [1] "MSFT" "AAPL"
```


xts object

- xts is a wide format. In contrast, dplyr/ggplot prefers long format.
- We have pivot_longer/pivot_wider to convert between long/wide format.
- Create xts object:
  - Put index aside, which is usually date
  - Store prices in columns.

```
library(xts)

# if df is a data frame/tibble.
# Date | V | GS
# To convert from tibble to xts obj
xobj <- xts(x=df[, -1, drop = FALSE], order.by = df[1])

# coredata(): returns a matrix from xts objects
core_data <- coredata(xobj)

# index: vector of Date/Time
index(xobj)
# Converts from xts to tibble
tibble(Time = index(xobj), as_tibble(coredata(xobj)))
xts2tb <- function(x) {
  tibble(Time = index(x), as_tibble(coredata(x)))
}
```

Get data from xts object

- xts has built-in support to filter date/time.

```
# What price history is stored here.
str(SPY)
## An 'xts' object on 2007-01-03/2020-10-13 containing:
##   Data: num [1:3470, 1:6] 142 141 141 141 141 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr [1:6] "SPY.Open" "SPY.High" "SPY.Low" "SPY.Close" ...
##   Indexed by objects of class: [Date] TZ: UTC
##   xts Attributes:
## List of 2
##  $ src    : chr "yahoo"
##  $ updated: POSIXct[1:1], format: "2020-10-14 15:36:40"
```

```
SPY2003 <- SPY["2003"]
SPY2 <- SPY["2003/2007"]
SPY3 <- SPY["2003-03-01/2007-07-01"]
SPY4 <- SPY["/2007-07-01"] # till
SPY5 <- SPY["2007-07-01/"] # from
SPY6 <- SPY["2007-07-01/", "SPY.High"]
SPY7 <- SPY["2007-07-01/", c("SPY.High", "SPY.Close")]
xts2tb(SPY7)
```


How to extract all components from S&P 500

We can use rvest package to crawl the wiki page.

```
library(rvest)

wikispx <- read_html('https://en.wikipedia.org/wiki/List_of_S%26P_500_companies')
currentconstituents <- wikispx %>%
  html_node('#constituents') %>%
  html_table(header = TRUE)

currentconstituents
```

S&P Changes

S&P 500 takes the 500 largest market cap companies. The list is updated periodically. Following code also extracts changes over the year. To have the list of constituents in the past, you can restore manually. (This was an exercise in Prep course).

```
spxchanges <- wikispx %>%
  html_node('#changes') %>%
  html_table(header = FALSE, fill = TRUE) %>%
  filter(row_number() > 2) %>% # First two rows are headers
  `colnames<-`(c('Date','AddTicker','AddName','RemovedTicker',
                 'RemovedName','Reason')) %>%
  mutate(Date = as.Date(Date, format = '%B %d, %Y'),
         year = year(Date),
         month = month(Date))
```


Quandl

```
library(Quandl)
library(tidyverse)

# Sign up with Quandl to get a token
# token_qd <- "xxxx"
Quandl.api_key(token_qd)
## You don't get SPY: SPDR 500 ETF from Quandl from free service.
## rates <- Quandl(c("EOD/SPY"), start_date="2000-01-01", end_date="2013-06-07")
## You don't get EOD US Stocks for free from Quandl from 2019
## rates <- Quandl(c("EOD/V"), start_date="2000-01-01", end_date="2013-06-07" )
```

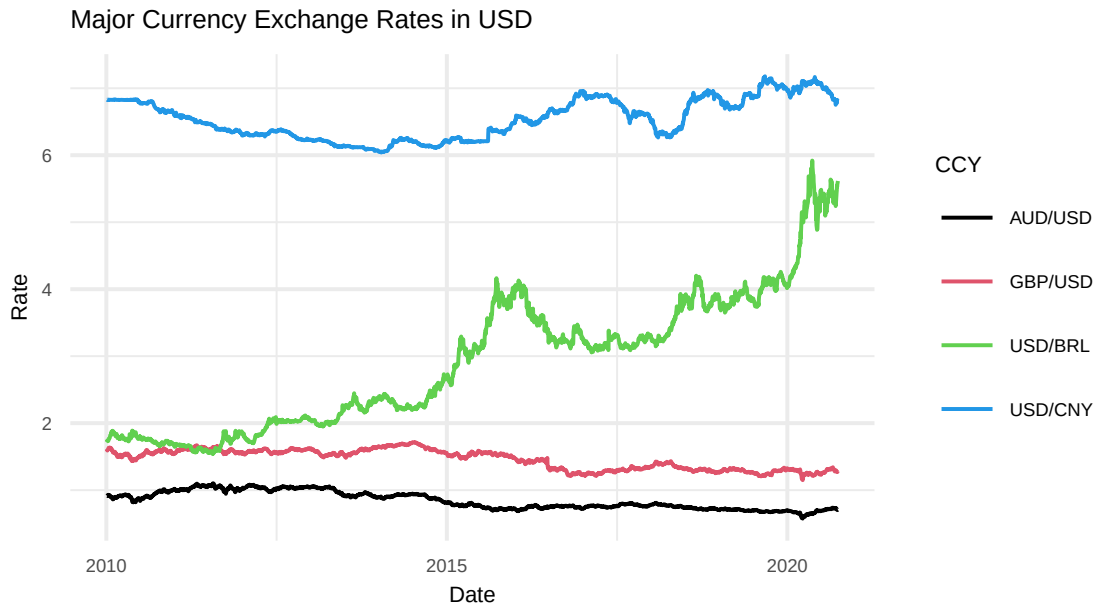
Quandl

```
library(Quandl)                    # Quandl package
library(ggplot2)                   # Package for plotting
library(tidyverse)                 # Package for reshaping data

Quandl.api_key(token_qd)                  # Authenticate your token
# Build vector of currencies
rates <- Quandl(c("FRED/DEXUSAL", "FRED/DEXBZUS", "FRED/DEXUSUK", "FRED/DEXCHUS"),
                start_date="2010-01-01",
                end_date = "2020-09-28")
colnames(rates) <- c("Date", "AUD/USD", "USD/BRL", "GBP/USD", "USD/CNY")
meltdf <- pivot_longer(rates, -Date, names_to = "CCY", values_to = "Rate")
```


Quandl - Result

```
ggplot(meltdf, aes(x = Date, y = Rate, colour = CCY, group = CCY)) +
  geom_line() +
  scale_colour_manual(values=1:22)+
  ggtitle("Major Currency Exchange Rates in USD") +
  theme_minimal() + theme(text = element_text(size=6))
```



dygraphs

dygraphs for xts https://rstudio.github.io/dygraphs/shiny.html

```
dygraphOutput("dygraph")
dygraph(oil_combined_xts, main = "Oil Prices: Historical and Forecast") %>%
  # Add the actual series
  dySeries("Actual", label = "Actual") %>%
  # Add the three forecasted series
  dySeries(c("Lo_95", "Forecast", "Hi_95"))
```


Quandl/Shiny/dygraph

- 51-quandl.R

Trading Game

- See ‘oil_lm.Rmd’ as reference
- Open a blank Rmd and let do it together.