Post2

Simple&Fast Web Data Tool: Shiny

Intro

Shiny App is a powerful tool to create web app for data engineers & scientists. It makes web visualization and interaction become very easy for data engineers. In the past, I really cannot tell the difference between R and python in their data processing ability. But after experiecing with shiny, I have to say R has a better visualization ability for its customers. I have been a web programmers for quite a long time, and I know how time-consuming it is to make website. If a data scienists has to spend a lot of time on web creation, it's going to lower their data analysis quality. I am a computer science major student and I have been through a lot of difficulities in creating websites before. It's a pain to spend a lot of time to build something that's only served to display some other contents. Therefore, in this Post, I'll make a comparsion between normal web development process and R web process and demonstrate some of Shiny's advantages.

First, let's understand why web visualization is important first. In fact, one of the advantage we have in Shiny App is that it gives us a better visualization than just a R plotted histogram. Consider the ugly histogram below:

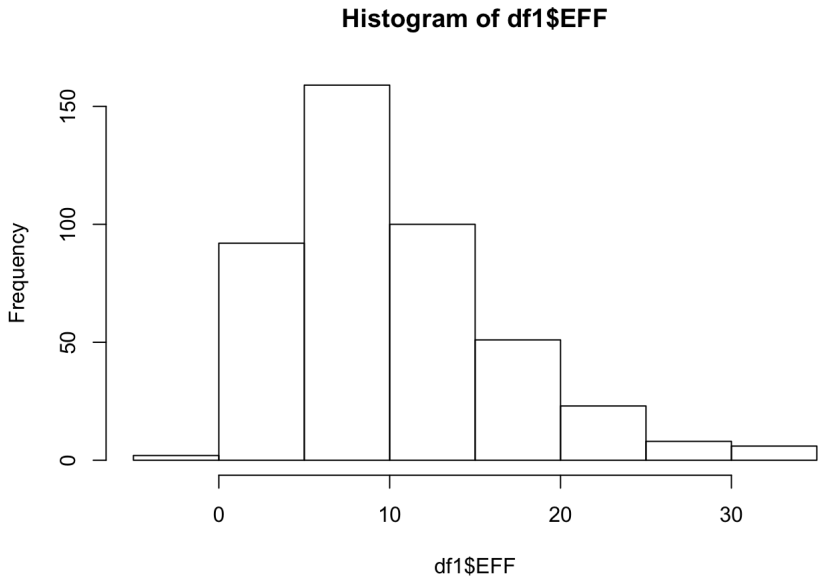
```
library(readr)
```

```
## Warning: package 'readr' was built under R version 3.3.2
```

```
# vector of data types (for each column)
col_types <- c(
    'character', 'character', 'factor', 'character', 'double','integer', 'integer', 'integer', 'integer','integer'
, 'integer','integer', 'integer','integer', 'integer','integer', 'integer','integer', 'integer','integer', 'intege
r','integer', 'integer','integer'
)

df1 <- read.csv(
    'nba2017-player-statistics.csv',
    header = TRUE,
    colClasses = col_types,
    sep = ","
)

df2 <- read_csv(
  'nba2017-player-statistics.csv',
   col_types  = list(col_character(), col_character(), col_factor(c("C", "PF", "PG", "SF", "SG")), col_character()
, col_double(), col_integer(), col_integer(),col_integer(),col_integer(),col_integer(),col_integer(),col_integer()
,col_integer(),col_integer(),col_integer(),col_integer(),col_integer(),col_integer(),col_integer(),col_integer(),c
ol_integer(),col_integer(),col_integer(),col_integer())
)
df1 <- data.frame(df1)
df2 <- data.frame(df2)
df1$Experience <- chartr("R", "0", df1$Experience)
df1<-transform(df1, Experience = as.integer(Experience))
df2$Experience <- chartr("R", "0", df2$Experience)
df2<-transform(df1, Experience = as.integer(Experience))

df1$Missed_FG <- df1$FGA - df1$FGM
df1$Missed_FT <- df1$FTA - df1$FTM
df1$PTS <-df1$Points3*3 + df1$Points2*2 + df1$FTM
df1$REB <-df1$OREB + df1$DREB
df1$MPG <- df1$MIN / df1$GP
df1$EFF <- (df1$PTS + df1$REB + df1$AST + df1$STL + df1$BLK - df1$Missed_FG - df1$Missed_FT - df1$TO) / df1$GP
summary(df1$EFF)
```

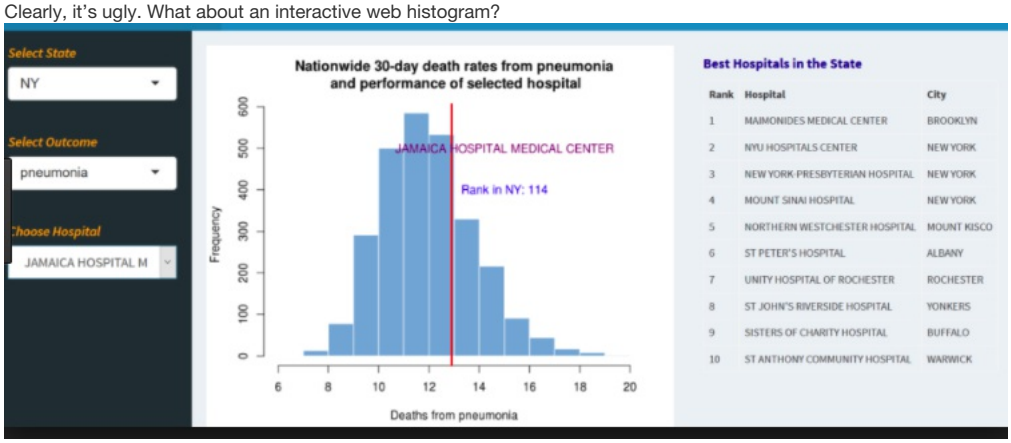
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -0.600   5.452   9.090  10.140  13.250  33.840
```

```
print (hist(df1$EFF),  main="Histogram of EFF", xlab="EFF")
```



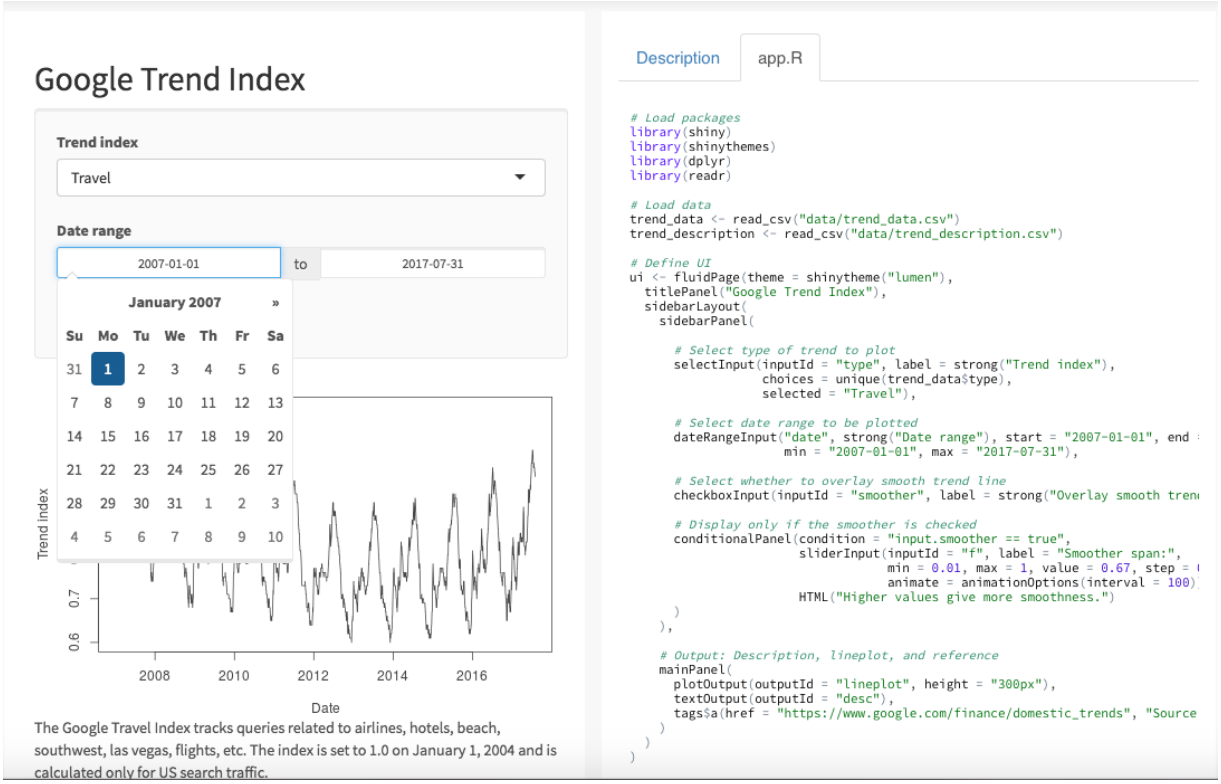

```
## $breaks
## [1] -5  0  5 10 15 20 25 30 35
##
## $counts
## [1]   2  92 159 100  51  23   8   6
##
## $density
## [1] 0.0009070295 0.0417233560 0.0721088435 0.0453514739 0.0231292517
## [6] 0.0104308390 0.0036281179 0.0027210884
##
## $mids
## [1] -2.5  2.5  7.5 12.5 17.5 22.5 27.5 32.5
##
## $xname
## [1] "df1$EFF"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

Clearly, it's ugly. What about an interactive web histogram?

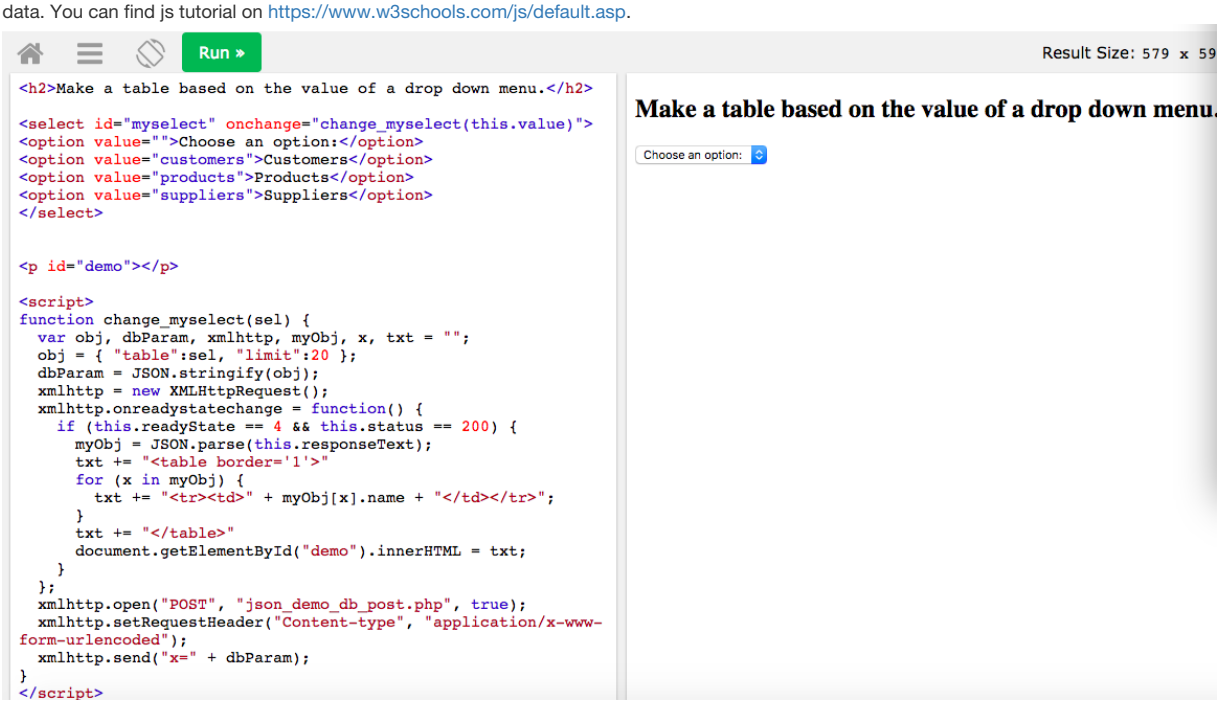


Much better right?!

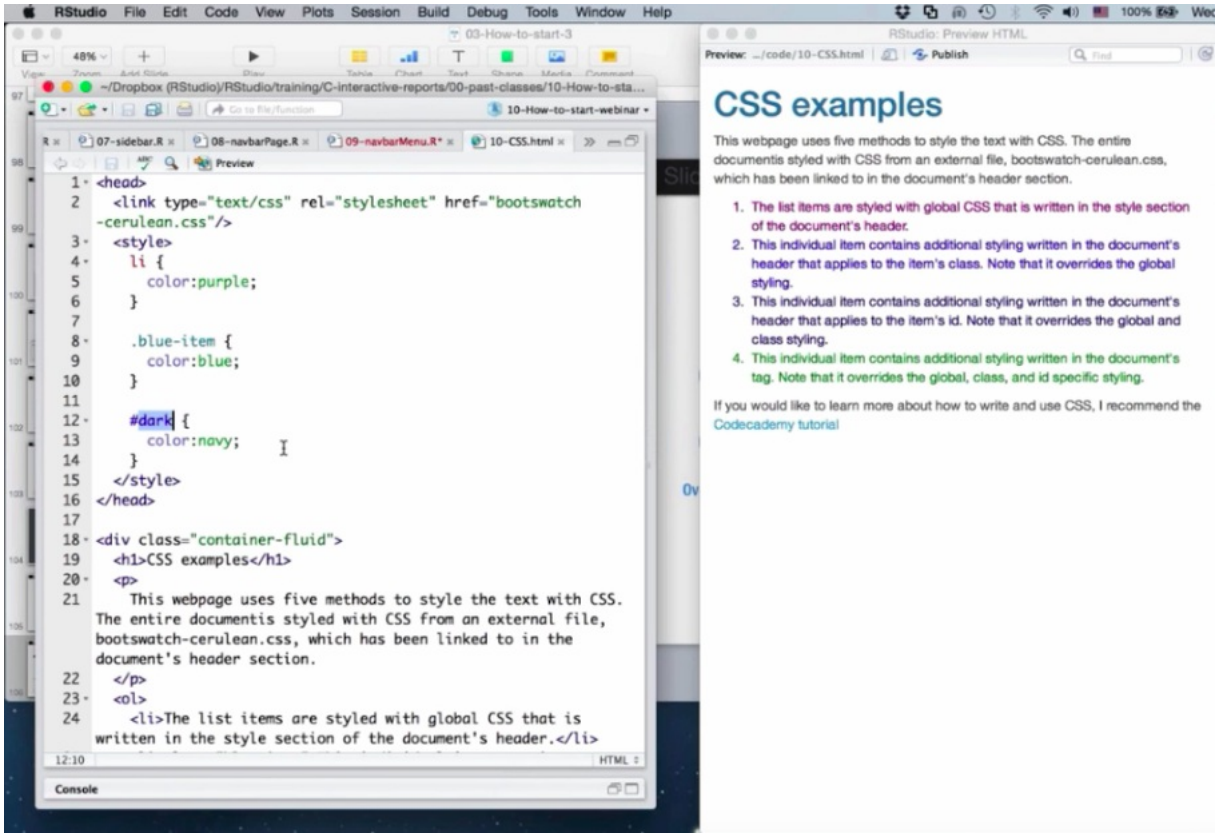
This is how a data analytics would normally create a interactive website using shiny package in R. This demo is from https://shiny.rstudio.com/. It has a nice option control as well as date range input fields. Also, the chart is very clear on data's relationship.



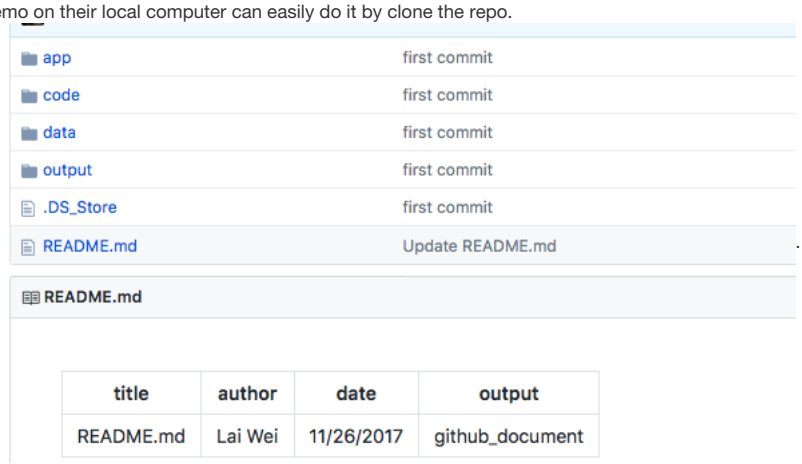
On the other hand, If a person tries to create a normal interactive website, he first needs to learn how to write Javascript code. As shown below, writing website codes could be very time consuming. Below is a snapshot of javascript code that merely creates a dropdown list, let alone import data. You can find js tutorial on https://www.w3schools.com/js/default.asp.



One may raise a question that if shiny app specifies everything, isn't the format very similar between each other? Well, the main concern for data analytics is to showcase their data analysis not the beauty layout. In addition，it's actually possible for data engineers to modify the css property, according to the video tutorial at 2:15:55 in https://vimeo.com/rstudioinc/review/131218530/212d8a5a7a/#t=1h11m36s.



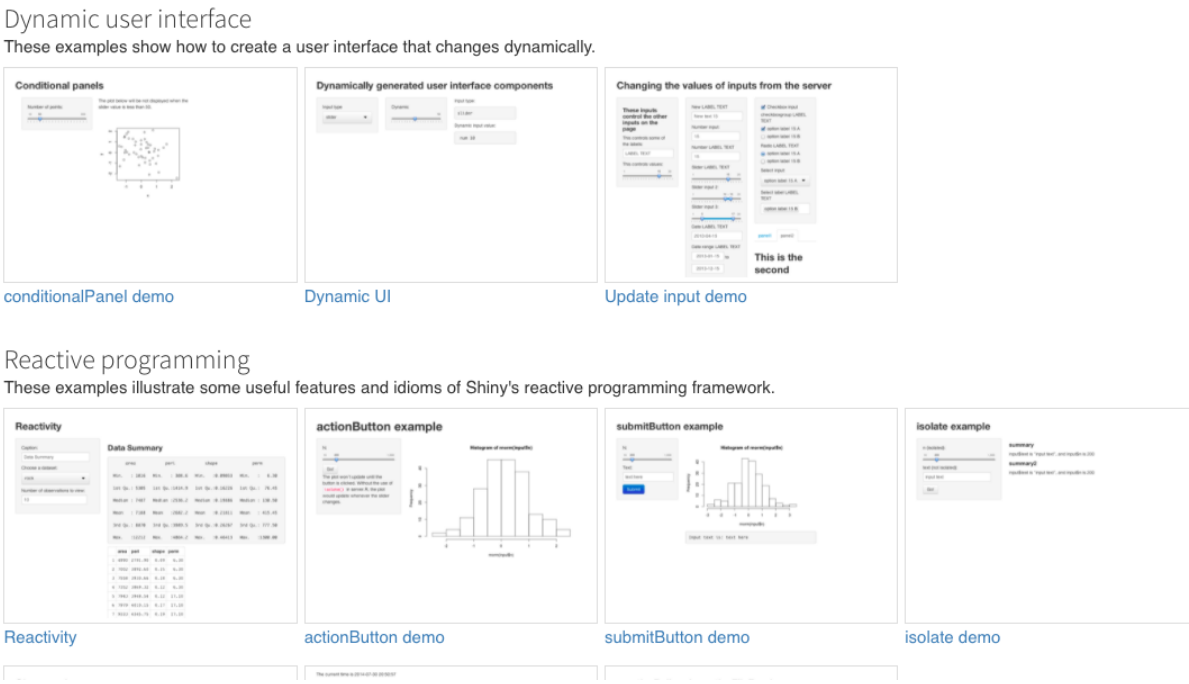
what's more, Shiny app allows you to easily share your data visualization as well as your data analysis work to your collagues. Because they're all integrated in the same R framework, all the files are R files. For example, this pic is from my github repository, and people who want to have a demo on their local computer can easily do it by clone the repo.



Above is a basic overview of the architure of shiny app and its advantages. Here I will give a more specific layout of its design. A shiny app is typically consist of 3 parts.

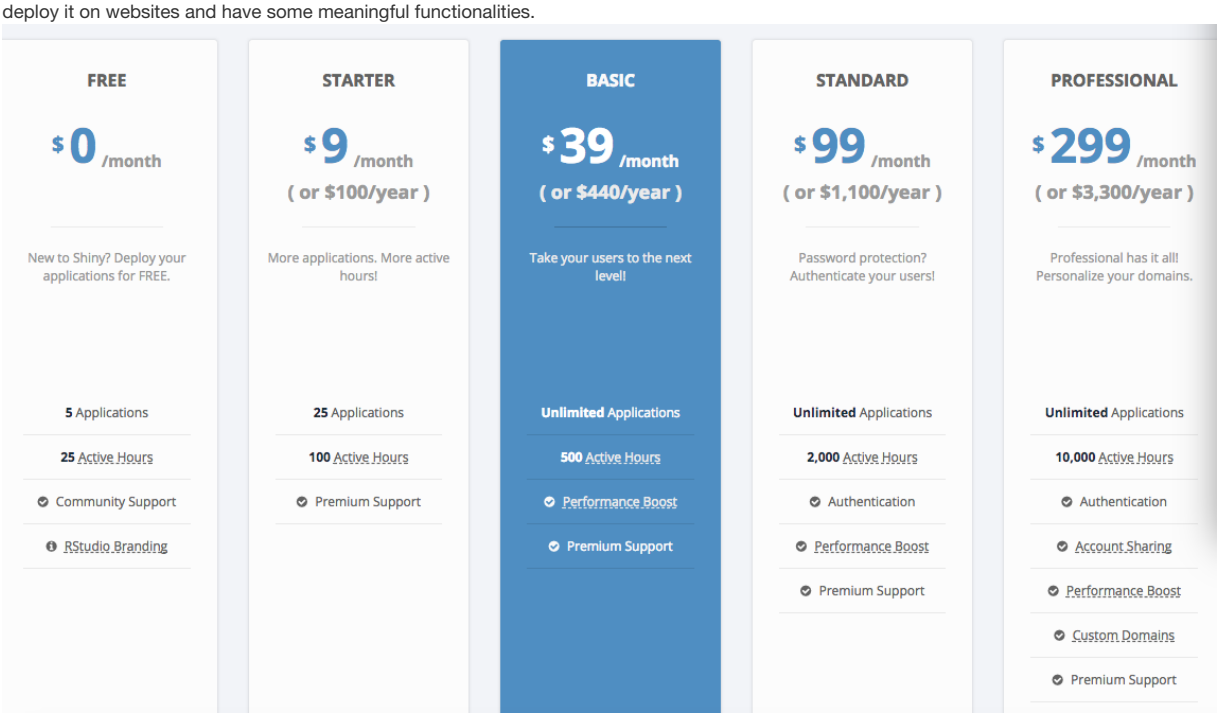
```{r}
library(shiny)

# Define UI ----
ui <- fluidPage(

)

# Define server logic ----
server <- function(input, output) {

}

# Run the app ----
shinyApp(ui = ui, server = server)
```

And it's very user-friendly. In fact, you can find many templates and inspirations from shiny gallery. By filling in some data path, you can easily demonstrate your work using the pre-built templates.



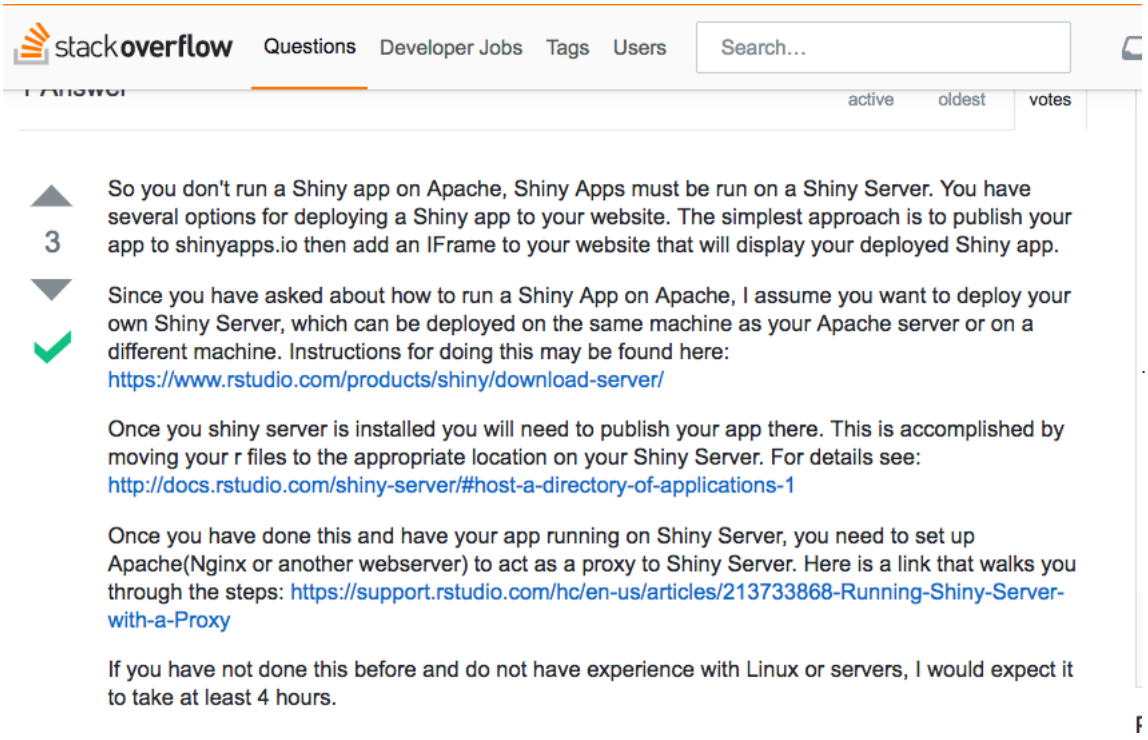
.

However, Shiny App is not 100% all good. For example, it's actually a non-free service. It charges certain amount of fees if you want to actually deploy it on websites and have some meaningful functionalities.

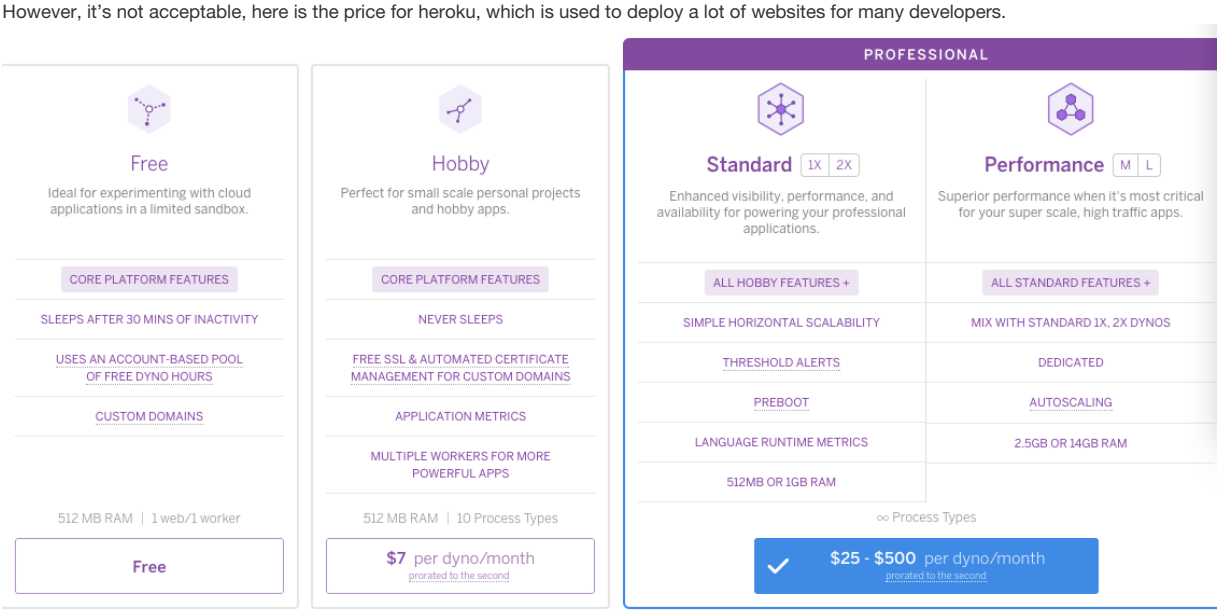


.

So can we run a shiny app on our Apache server? The answer seems to be no according to the answer found on stackoverflow.



However, it's not acceptable, here is the price for heroku, which is used to deploy a lot of websites for many developers.



.

So it seems like shiny's pricing isn't that bad.

While shiny has less control for programmers compared to normal javascript website, it is still a very convenient tool for data engineers or data researchers to show their work.

Reference: https://shiny.rstudio.com/

https://www.shinyapps.io/

https://vimeo.com/rstudioinc/review/131218530/212d8a5a7a/#t=1h11m36s

https://www.heroku.com/pricing?c=70130000001xDpdAAE&gclid=EAIaIQobChMIwOvSmprt1wIVklp-Ch09tQ14EAAYASABEgJOyPD_BwE

https://www.w3schools.com/js/default.asp

https://shiny.rstudio.com/gallery/

https://stackoverflow.com/questions/43527041/run-r-shiny-app-on-apache-server