post02-Yichi-Zhang

Creating Interactive Maps Using Shiny App and Leaflet R Package.

Introduction

Launching a interactive map within R has been possible for some years, and a number of packages for doing this are available, including: RgoogleMaps, an interface to the Google Maps api leafletR, an early package for creating Leaflet maps with R rCharts. In this tutorial we will first learn how to new RStudio-supported leaflet R package, and then use both leaflet and Shiny App to create a small project, which will be an interactive map that shows the national parks in Hawaii.

Motivation

We've learned all kinds of tools in R that help us to visualize data. Even though I consider Shiny App to be the most time consuming one, after all of the hard works I've done in homeworks and labs, I actually found the process of generating and learning new things from Shiny App really enjoyable. Today, I will explore more advanced Shiny App skills in order to create more interactive visualizations, and use the package "Leaflet" to help achieving this goal. Since I am going to spend my Christmas holidays in Hawaii, it really motivates me to explore the national parks in Hawaii and develop an interactive map as my travel guide. See, that's why we learn R language, it can actually be used in our every day life.

Overall Content

The post contains two big parts. The first part of the post will explore the new R package leaflet, and show how string basics and for loops are practical in using leaflet. The second part is mainly about creating the final shiny app that shows the interactive map. All the codes in this post are reproducible in an Rmd file you just need to create code chunks using ``` `{r}` , and all the dataset of the map information are available online or included in R dataset which can be used directly in R. Additionally, all of the codes I provide don't include the last part where you print the name of the map and generate it, it is because generating interactive mapshot and html appshot of Shiny app objects are not yet supported in Rstudio.

Leaflet R Package

leaflet is supported by RStudio, and it features interactive panning/zooming, and can be created right from the R console or RStudio. First, I will show you the basic usage of the package first.

1) Basic steps

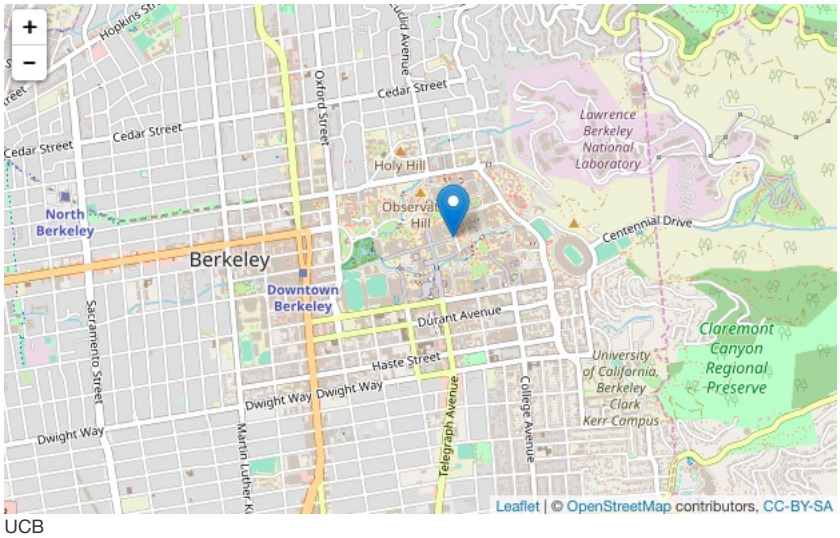
Shiny is designed for fully interactive visualization. First of all, install the package using `install.packages("leaflet")` and load it to Rstudio with `library(leaflet)`. Then we can create a map widget by calling `leaflet()`. The we can add layers to the map by using layer functions to modify the map widget. Repeat step 2 as desired. The last step is to print the map widget to display it. The most important thing here is that in order to locate the targeted place, longitude and latitude of the location is need. Here is a code example of showing the location of UC Berkeley on an interactive map using leaflet:

```
#install.packages("leaflet")
library(leaflet)

## 'leaflet' objects
ucb <- leaflet() %>%
  addTiles() %>%  # Add default OpenStreetMap map tiles
  addMarkers(lng = -122.2585, lat = 37.8719) #lat and lng are obtained from web source
```

Don't forget to print `ucb` to generate the map. When you try to reproduce this map you will realize that since the map involves interactive panning/zooming, it can't be knitted to html in Rmd. Therefore, the following picture is a screen shot dadicating the apparence of the map, try to do it in yout Rstudio and play around with it!

Interactive Map of UC Berkeley



UCB

2) Leaflet Package with for loop

GADM (database of Global Administrative Areas) is a spatial database of the location of the world's administrative areas for use in GIS and similar

software. You can download a free program such as Q-GIS or DIVA-GIS. The RData files can be used in R with the 'sf' and 'sp' package loaded. Here is an example of leaflet using "for loop" to coloring selected locations on an world map using GADM data:

```
#install.packages("sp")
#install.package("sf")
#install.package("mapview")
#install.package("raster")
library(sp)
library(sf)
```

```
## Warning: package 'sf' was built under R version 3.4.2
```

```
## Linking to GEOS 3.6.1, GDAL 2.1.3, proj.4 4.9.3
```

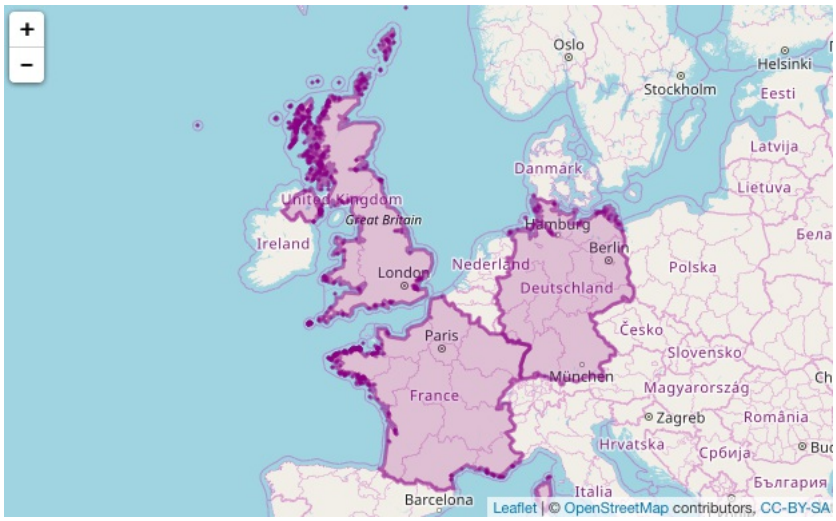
```
library(mapview)
```

```
## Warning: package 'mapview' was built under R version 3.4.2
```

```
library(raster)
#Select France, UK and Germany on the map
countries_1 <- c('FRA', 'GBR', 'DEU')
#select data set GADM
dat_list = lapply(countries_1, function(i) {
  st_as_sf(getData("GADM", country = i, level = 0))
  })
#Generate leaflet map
worldmap = leaflet() %>%
  addTiles()
#Using for loop to create colored circles
for (i in dat_list) {
  worldmap = mapview::addFeatures(
    map = worldmap,
    data = i,
    weight = 3,
    fillColor = 'purple',
    color = 'purple'
    )
}
```

Don't forget to print `worldmap` to generate the map.

Interactive Map with Colored Regions



Countries

This is the graph that you should obtain from the code. the three country with purple circles is what we select with the loop.

3) Leaflet package with string basic

Now that we are familiar with the basic process of leaflet in generating the map, let's create a more interactive map with string basic. Here is an example of using string basic especially "paste0" to insert a popup image that is available online in the map by pasting the URL into the leaflet function:

```
#The URL of the popup image
file <- 'http://www.beach-backgrounds.com/wallpapers/the-inspiring-wallpaper-of-the-waialae-beach-honolulu-hawaii-128x128-616.jpg'
#create a map of "honolulu"
honolulu <- leaflet() %>%
  addTiles %>%
  addMarkers(
  lng = -157.8583,
  lat = 21.3069,
  popup = paste0("<img src=", file, ">"), #insert the popup image
  popupOptions = popupOptions(maxWidth = "100%", closeOnClick = TRUE)
  ) # make the popupimage fit the size of the map
```

Don't forget to print the map using `honolulu` to see map.

Interactive Map of Honolulu with a Popup Image

honolulu

Tab on the marker to show the popup image, you will see the same image shown above.

Creating an interactive map using both Shiny App and leaflet package

The Leaflet package includes powerful and convenient features for integrating with Shiny applications. Most Shiny output widgets are incorporated into an app by including an output for the widget in the UI definition, and using a render function such as `renderPlot()` in the server function. Leaflet maps are the same, in the UI you call `leafletOutput()`, and on the server side you assign a `renderLeaflet()` call to the output. Inside the "renderLeaflet" expression, you return a leaflet map object. The map I'm intended to create will include two parts, the sidebar containing one selectinput indicating whether the national park is free or not. It aims to filter desired national parks.

1) Data frame

Here is the code to create the data frame:

```
#create data frame
nparks <- data.frame(
  Names = c(
  "Ala Kahakai",
  "Haleakala",
  "Hawai'i Volcanoes",
  "Kalaupapa",
  "Kaloko-Honokohau",
  "Pu'uhonua O Honaunau",
  "World War II Valor in the Pacific"
  ) ,
  Fees = c(
  "Free",
  "$12/person",
  "$12/person",
  "Free",
  "Free",
  "$3/person",
  "Free"
  ) ,
  Islands = c(
  "Hawai'i",
  "Maui ",
  "Hilo",
  "Hawai'i",
  "Moloka'i",
  "Kawaihae",
  "O'ahu"
  ),
  Lat = c(
  19.686807,
  20.709692,
  19.419370,
  21.189301,
  19.678706,
  19.421539,
  21.367536
  ),
  Long = c(
  -156.032572,
  -156.253515,
  -155.288497,
  -156.981805,
  -156.021700,-155.910525,
  -157.938595
  ),
  #Longitute and latitude are necessary to locate the parks in leaflet

  #Files use wab image and the URL is included in the data frame.
  files = c(
  "https://pbs.twimg.com/media/DPrgyDMV4AM5r-o.jpg",
  "https://pbs.twimg.com/media/DP05MkyUQAEApyl.jpg",
  "https://pbs.twimg.com/media/DPrgyDaUQAE6Y45.jpg",
  "https://pbs.twimg.com/media/DPrgyDOV4AAIyxw.jpg",
  "https://pbs.twimg.com/media/DP05WTPUMAUD8sb.jpg",
  "https://pbs.twimg.com/media/DP05PBvUMAAeyUN.jpg",
  "https://pbs.twimg.com/media/DPrgyDZU8AAHmqc.jpg"
  )
  )

  #Clean up the data frame
  rownames(nparks) <- nparks[, 1]
  nparks[, 1] <- NULL
  nparks
```

```
##                                     Fees  Islands      Lat      Long
## Ala Kahakai                         Free  Hawai'i 19.68681 -156.0326
## Haleakala                     $12/person    Maui  20.70969 -156.2535
## Hawai'i Volcanoes             $12/person     Hilo 19.41937 -155.2885
## Kalaupapa                           Free  Hawai'i 21.18930 -156.9818
## Kaloko-Honokohau                    Free Moloka'i 19.67871 -156.0217
## Pu'uhonua O Honaunau          $3/person Kawaihae 19.42154 -155.9105
## World War II Valor in the Pacific   Free    O'ahu 21.36754 -157.9386
##                                                                   files
## Ala Kahakai                   https://pbs.twimg.com/media/DPrgyDMV4AM5r-o.jpg
## Haleakala                     https://pbs.twimg.com/media/DP05MkyUQAEApyl.jpg
## Hawai'i Volcanoes             https://pbs.twimg.com/media/DPrgyDaUQAE6Y45.jpg
## Kalaupapa                     https://pbs.twimg.com/media/DPrgyDOV4AAIyxw.jpg
## Kaloko-Honokohau              https://pbs.twimg.com/media/DP05WTPUMAUD8sb.jpg
## Pu'uhonua O Honaunau          https://pbs.twimg.com/media/DP05PBvUMAAeyUN.jpg
## World War II Valor in the Pacific https://pbs.twimg.com/media/DPrgyDZU8AAHmqc.jpg
```

2) Ui and Server functions for Shiny App

Now after we creating the data frame, we can start to put together everything and create the shiny app. Let's start with create the ui part

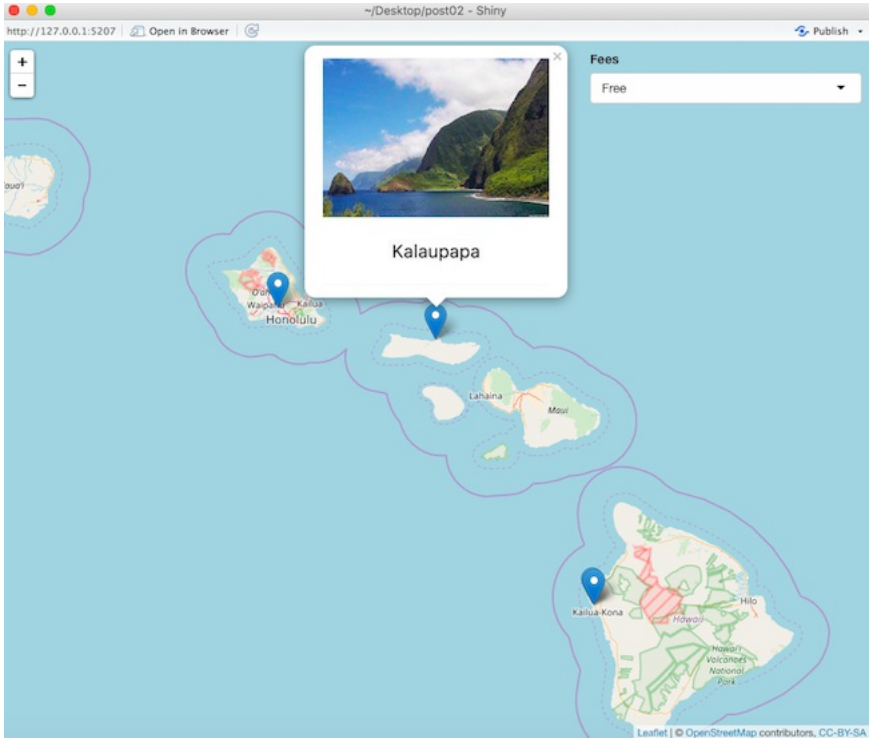
```
#ui part of the shiny app
library(shiny)
library(leaflet)
#ui function with selectInput Fees
ui <- bootstrapPage(
  tags$style(type = "text/css", "html, body {width:100%;height:100%}"),
  leafletOutput("mymap", width = "100%", height = "100%"),
  absolutePanel( top = 10, right = 10,
                 selectInput("fees","Fees",
                             choices = c("Free", "$12/person", "$3/person"),
                             selected = "Free"
                             )
                 )
  )


server <- function(input, output, session) {
  output$mymap <- renderLeaflet({
#using if function to filter the national parks shown
    if (input$fees == "Free") {
      leaflet(data = nparks[nparks$Fees == "Free", ]) %>%
        addTiles() %>%
        addMarkers( ~ Long, ~ Lat,
                    popup = paste0("<img src=", nparks[nparks$Fees == "Free", 5], ">"),
                    popupOptions = popupOptions(maxWidth = "100%", closeOnClick = TRUE)
                    )

      } else if (input$fees == "$12/person") {
        leaflet(data = nparks[nparks$Fees == "$12/person", ]) %>%
          addTiles() %>%
          addMarkers( ~ Long, ~ Lat,
                      popup = paste0("<img src=", nparks[nparks$Fees == "$12/person", 5], ">"),
                      popupOptions = popupOptions(maxWidth = "100%", closeOnClick = TRUE)
                      )
        } else if (input$fees == "$3/person") {
          leaflet(data = nparks[nparks$Fees == "$3/person", ]) %>%
            addTiles() %>%
            addMarkers( ~ Long, ~ Lat,
                        popup = paste0("<img src=", nparks[nparks$Fees == "$3/person", 5], ">"),
                        popupOptions = popupOptions(maxWidth = "100%", closeOnClick = TRUE)
            )
          }
  })
}
```

Don't forget to include `shinyApp(ui, server)`. The reason why I don't include the code in the code chunk is that when generating html appshot of Shiny app objects is not yet supported.

Interactive Map inside Shiny App



shinyapp

The reason why I don't include the code in the code chunk is that when generating html appshot of Shiny app objects is not yet supported.

Take Home Message

Making a shiny app is actually fun and easy right? It's not only useful in analyzing academic report we studied in class, it could actually be implemented in our daily life. What's more, different R packages or tools can actually be used in each other,; here, I showed the example of combining two applications into one project. What I also want to convey is that interactive visualization is an important part of R language , it is an area with lots of potentials. Last but not least, concepts like "for loop", basic string manipulation can be implemented in all kinds of packages, and we really need to have a strong ability to use these basic and fundamental functions so that we can expand our knowledge in a wider range in the future. I really enjoyed this semester's learning of R language, how about you? Thanks for reading my post, and good luck on your finals!

Reference

- Leaflet marker functions: https://stackoverflow.com/questions/36433899/image-in-r-leaflet-marker-popups/36434237
- Mapview popup functions: https://environmentalinformatics-marburg.github.io/mapview/popups/html/popups.html
- Package 'leaflet': https://cran.r-project.org/web/packages/leaflet/leaflet.pdf
- Get Latitude and Longitude: https://www.latlong.net/
- Using leaflet in Shiny: https://rstudio.github.io/leaflet/shiny.html
- Maps in Shiny: https://www.showmeshiny.com/category/visualization/maps/
- Introduction of leaflet: https://rstudio.github.io/leaflet/