STARTING APP

# ---------------------------------------------------------------------------- #

# ACTIVITY 2.3d map\_4 SERVER

# ---------------------------------------------------------------------------- #

##Load libraries

library(shiny)

library(leaflet)

library(lubridate)

library(dplyr)

library(rgdal)

library(rgeos)

# Load in the raw data

raw\_data <- read.csv("data/raw\_data.csv", stringsAsFactors=FALSE)

# Add year and decimal date to data

leaflet\_data <- raw\_data %>%

mutate(year = substr(date, 1,4), date=ymd(date), date\_decimal = decimal\_date(date))

## Create a colour palette for points

palette <- c("#231D51", "#178B8B", "#63C963", "#FFE31D")

## Load region shapefile

regions <- readOGR("data/TZ\_Region\_2012","TZ\_Region\_2012")

## Load protected areas shapefile

PAs <- readOGR("data/TZprotected\_areas","TZprotected\_areas")

#------------------------------------------------------------------------------#

# Define server logic

shinyServer(function(input, output) {

## Subset data based on inputs

leaflet\_data\_sub<- reactive({

leaflet\_data %>%

filter(species %in% input$species)

})

## Create text pop-up information for each point in subsetted data

popupInfo <- reactive({

paste("Date: ", leaflet\_data\_sub()$date, "<br>",

"Species: ", leaflet\_data\_sub()$species, "<br>",

"Age: ", leaflet\_data\_sub()$age, "<br>",

"Sex: ", leaflet\_data\_sub()$sex, "<br>",

sep = " ")

})

# Get point colours based on chosen variable

pal <- reactive({

colourby\_col <- ifelse(input$colourby!="date",input$colourby,"date\_decimal")

if(input$colourby %in% c("species","sex")){

colorFactor(palette, domain = sort(unique(leaflet\_data[,colourby\_col])))

}else if(input$colourby %in% c("date","age")){

colorNumeric(palette, range(leaflet\_data[,colourby\_col]))

}

})

## Render map

output$mymap <- renderLeaflet({

## Initialise map with tile. Set central point of viewing window and initial amount of zoom.

m <- leaflet() %>%

addProviderTiles("Stamen.Terrain") %>%

setView(c(gCentroid(regions)@coords)[1], c(gCentroid(regions)@coords)[2], zoom = 6)

## Add selected shapefiles

if("regions" %in% input$shapefiles){

m <- m %>%

addPolygons(data=regions,color="black",fillColor = "white",

label=regions$Region\_Nam, weight=1, fillOpacity=0.7)}

if("protected areas" %in% input$shapefiles){

m <- m %>%

addPolygons(data=PAs,color="transparent",fillColor = "tomato",

weight=1, fillOpacity=0.7)}

## Add coloured points and legend

colourby\_col <- ifelse(input$colourby!="date",input$colourby,"date\_decimal")

m %>%

addCircles(data=leaflet\_data\_sub(),lng=~leaflet\_data\_sub()$x,lat=~leaflet\_data\_sub()$y,

color = pal()(leaflet\_data\_sub()[,colourby\_col]),

opacity=1, fillOpacity=1, popup = popupInfo()) %>%

addLegend(position = "bottomright", title = input$colourby,

pal = pal(), values = leaflet\_data[,colourby\_col], opacity=1,

labFormat = labelFormat(big.mark = ""))

})

})

STARTING APP

# ---------------------------------------------------------------------------- #

# ACTIVITY 2.3d map\_4 UI

# ---------------------------------------------------------------------------- #

# Load libraries

library(shiny)

library(leaflet)

library(shinyWidgets)

library(dplyr)

library(lubridate)

# Load in the raw data

raw\_data <- read.csv("data/raw\_data.csv", stringsAsFactors=FALSE)

# Tranform dates from characters to date objects

leaflet\_data <- raw\_data %>% mutate(date=ymd(date))

# Get the unique names of the species for the drop down menu

all\_species <- unique(leaflet\_data$species)

#------------------------------------------------------------------------------#

# Define UI for application

shinyUI(fluidPage(

# Application title

titlePanel("Day 2 - Map\_4"),

# Add a line break

br(),

# Add text section

h4("A map app, where we can select the species to display and the variable to colour points by.

Optional display of region and protected area shapefiles."),

# Add a line break

br(),

sidebarLayout(

# Sidebar containing the widgets

sidebarPanel(

# Drop down menu to choose variable by which points will be coloured

selectInput(inputId="colourby", label="Colour Cases By:",

choices = c("species","date","sex","age"),

selected="species"),

br(),

# Menu for selecting which species to display

pickerInput(inputId = "species", label = "Species:",

sort(all\_species), selected= all\_species, # Use sort to get names in alphabetical order

options = list(`actions-box` = TRUE,`live-search` = TRUE), multiple = T),

br(),

# Checkboxes for choosing shapefiles to be displayed

checkboxGroupInput("shapefiles", label = "Select background polygons:",

choices = c("regions", "protected areas"),

selected = c("regions"))

),

# Show a plot of the map

mainPanel(

leafletOutput("mymap",width=800,height=500)

)

)

))

COMPLETED APP

# ---------------------------------------------------------------------------- #

# ACTIVITY 2.3d map\_4 SERVER

# ---------------------------------------------------------------------------- #

##Load libraries

library(shiny)

library(leaflet)

library(lubridate)

library(dplyr)

library(rgdal)

library(rgeos)

# Load in the raw data

raw\_data <- read.csv("data/raw\_data.csv", stringsAsFactors=FALSE)

# Add year and decimal date to data

leaflet\_data <- raw\_data %>%

mutate(year = substr(date, 1,4), date=ymd(date), date\_decimal = decimal\_date(date))

## Create a colour palette for points

palette <- c("#231D51", "#178B8B", "#63C963", "#FFE31D")

## Load region shapefile

regions <- readOGR("data/TZ\_Region\_2012","TZ\_Region\_2012")

## Load protected areas shapefile

PAs <- readOGR("data/TZprotected\_areas","TZprotected\_areas")

#------------------------------------------------------------------------------#

# Define server logic

shinyServer(function(input, output) {

## Subset data based on inputs

leaflet\_data\_sub<- reactive({

leaflet\_data %>%

filter(date>input$date[1] & date<input$date[2] & species %in% input$species)

})

## Create text pop-up information for each point in subsetted data

popupInfo <- reactive({

paste("Date: ", leaflet\_data\_sub()$date, "<br>",

"Species: ", leaflet\_data\_sub()$species, "<br>",

"Age: ", leaflet\_data\_sub()$age, "<br>",

"Sex: ", leaflet\_data\_sub()$sex, "<br>",

sep = " ")

})

# Get point colours based on chosen variable

pal <- reactive({

colourby\_col <- ifelse(input$colourby!="date",input$colourby,"date\_decimal")

if(input$colourby %in% c("species","sex")){

colorFactor(palette, domain = sort(unique(leaflet\_data[,colourby\_col])))

}else if(input$colourby %in% c("date","age")){

colorNumeric(palette, range(leaflet\_data[,colourby\_col]))

}

})

## Render map

output$mymap <- renderLeaflet({

## Initialise map with tile. Set central point of viewing window and initial amount of zoom.

m <- leaflet() %>%

addProviderTiles("Stamen.Terrain") %>%

setView(c(gCentroid(regions)@coords)[1], c(gCentroid(regions)@coords)[2], zoom = 6)

## Add selected shapefiles

if("regions" %in% input$shapefiles){

m <- m %>%

addPolygons(data=regions,color="black",fillColor = "white",

label=regions$Region\_Nam, weight=1, fillOpacity=0.7)}

if("protected areas" %in% input$shapefiles){

m <- m %>%

addPolygons(data=PAs,color="transparent",fillColor = "tomato",

weight=1, fillOpacity=0.7)}

## Add coloured points and legend

colourby\_col <- ifelse(input$colourby!="date",input$colourby,"date\_decimal")

m %>%

addCircles(data=leaflet\_data\_sub(),lng=~leaflet\_data\_sub()$x,lat=~leaflet\_data\_sub()$y,

color = pal()(leaflet\_data\_sub()[,colourby\_col]),

opacity=1, fillOpacity=1, popup = popupInfo()) %>%

addLegend(position = "bottomright", title = input$colourby,

pal = pal(), values = leaflet\_data[,colourby\_col], opacity=1,

labFormat = labelFormat(big.mark = ""))

})

})

COMPLETED APP

# ---------------------------------------------------------------------------- #

# ACTIVITY 2.3d map\_4 UI

# ---------------------------------------------------------------------------- #

# Load libraries

library(shiny)

library(leaflet)

library(shinyWidgets)

library(dplyr)

library(lubridate)

# Load in the raw data

raw\_data <- read.csv("data/raw\_data.csv", stringsAsFactors=FALSE)

# Tranform dates from characters to date objects

leaflet\_data <- raw\_data %>% mutate(date=ymd(date))

# Get the unique names of the species for the drop down menu

all\_species <- unique(leaflet\_data$species)

#------------------------------------------------------------------------------#

# Define UI for application

shinyUI(fluidPage(

# Application title

titlePanel("Day 2 - Map\_4"),

# Add a line break

br(),

# Add text section

h4("A map app, where we can select the species to display and the variable to colour points by. Optional display of region and protected area shapefiles. This app now also has a slider to select the date range of the points displayed"),

# Add a line break

br(),

sidebarLayout(

# Sidebar containing the widgets

sidebarPanel(

# Slider for date selection

sliderInput(inputId = "date", label = "Date:",

min = min(leaflet\_data$date), max =max(leaflet\_data$date),

value=c(min(leaflet\_data$date), max(leaflet\_data$date)),

timeFormat="%b %Y"),

br(),

# Drop down menu to choose variable by which points will be coloured

selectInput(inputId="colourby", label="Colour Cases By:",

choices = c("species","date","sex","age"),

selected="species"),

br(),

# Menu for selecting which species to display

pickerInput(inputId = "species", label = "Species:",

sort(all\_species), selected= all\_species, # Use sort to get names in alphabetical order

options = list(`actions-box` = TRUE,`live-search` = TRUE), multiple = T),

br(),

# Checkboxes for choosing shapefiles to be displayed

checkboxGroupInput("shapefiles", label = "Select background polygons:",

choices = c("regions", "protected areas"),

selected = c("regions"))

),

# Show a plot of the map

mainPanel(

leafletOutput("mymap",width=800,height=500)

)

)

))