TIF Shiny App 2022

This PDF serves as a document to aid future TIF Illuminators, or anyone with basic coding skills wishing to visualize future TIF data, in creating or updating the Shiny App function created by the 2023 TIF Illuminators team. The code for the app will be presented in code chunks below with detailed explanations of what the code will be accomplishing in plain text above. If any issues arise feel free to use the following contact information: Andres Meza email - meza.a.e.1293@gmail.com phone number - 773-666-3679

Loading in the neccessary libraries

```
library(shiny)
library(sf)
library(classInt)
library(leaflet)
library(tidyverse)
library(viridisLite)
library(scales)
library(gt)
library(htmltools)
library(shinythemes)
library(shinyWidgets)
library(rlang)
```

Reading in the updated csv. shapefiles from your local network and transforming them to be able to be used for map application Note: In st_read function make sure the pathway is where you are pulling the data from either locally or from a webpage

```
Sf2022 <- st_read("~/Desktop/Sf2022/chiTifBoundaries_2022")
```

```
## Reading layer 'chiTifBoundaries' from data source
##   '/Users/andresmeza/Desktop/Sf2022/chiTifBoundaries_2022' using driver 'ESRI Shapefile'
## Simple feature collection with 131 features and 28 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -87.79527 ymin: 41.64456 xmax: -87.52453 ymax: 42.01781
## Geodetic CRS:  WGS84(DD)
```

```
Sf2022 <- st_transform(Sf2022, crs = "+proj=longlat +datum=WGS84")
```

Reading in updated csv. file containing TIF information for the current year in a format consistent with the programs previous years

```
tif2022 <- read.csv("~/Desktop/2022_out.csv")
```

Merging the shapefile with the file containing current years TIF information

```
Sf2022 <- Sf2022 %>%
  inner_join(tif2022,by="tif_number")
```

Creating the table that will be displayed on the Shiny app

```
gt_tif_2022 <-
  tif2022 %>%
  gt() %>%
  fmt_currency(columns=c(property_tax_extraction,cumulative_property_tax_extraction,
                         transfers_in,cumulative_transfers_in,expenses,
                         fund_balance_end,transfers_out,distribution,admin_costs,
                         finance_costs),decimals=0) %>%
  cols_hide(columns=tif_year) %>%
  cols_label(tif_name="TIF Name",start_year="Start Year",end_year="End Year",
             tif_number="TIF #",property_tax_extraction="Property Tax Extraction",
             cumulative_property_tax_extraction="Cumulative Property Tax Extraction",
             transfers_in="Transfers In",
             cumulative_transfers_in="Cumulative Transfers In",expenses="Expenses",
             fund_balance_end="Fund Balance End",transfers_out="Transfers Out",
             distribution="Distribution",
             admin_costs="Administrative Costs",finance_costs="Finance Costs",
             bank="Bank") %>%
  opt_interactive(use_compact_mode=TRUE) %>%
  tab_header(title=tagList(tags$div(style=css(`text-align`="center"),
                                    HTML(local_image("~/Desktop/TIF_Illuminators.jpeg",height=px(65))))
                           tags$div("Chicago TIF Districts 2022"))) %>%
  cols_width(contains("bank") ~ px(200),ends_with("name") ~ px(250),starts_with("cumulative") ~ px(200)
```

Inputs to create the Shiny App

```
ui <- fluidPage(
  theme = shinytheme("cosmo"),
  setBackgroundColor(
    color = c("#FFFFFF", "#B3DDF2", "#FF0000"),
    gradient = "linear",
    direction = "right"
  ),
  tags$h1("Chicago TIF Districts 2022"),
  fluidRow(
    column(
      width = 4,
      varSelectInput("variable","Select Variable:",data=tif2022[,6:15],
                     selected="property_tax_extraction")),
    column(
      width = 8,leafletOutput(outputId = "map"))),
  gt_output(outputId = "table")
)
```

Shiny App server code

```
server <- function(input,output, session){
```

```
  output$map <- renderLeaflet({
    # Using Jenks natural breaks - similar to ArcGIS
    breaks_j <- with(Sf2022,classIntervals(inject(!!input$variable),n=7,style="jenks"))
    breaks_j_color <- findColours(breaks_j,plasma(7))
    p_popup <- paste0("<strong>Variable Selected: </strong>","<br>",
                      Sf2022$tif_name,"<br>",
                      with(Sf2022,inject(dollar(!!input$variable))))

    leaflet(Sf2022) %>% addPolygons(stroke=FALSE,
                                    fillColor=~breaks_j_color,
                                    fillOpacity=0.75,smoothFactor=0.5,
                                    popup=p_popup,group="Chicago") %>%
      addTiles(group="OSM") %>%
      addLegend("topright",
                colors=plasma(7),
                labels=paste0("up to ",dollar(breaks_j$brks[-1])),
                title="Variable Selected:") %>%
      addLayersControl(baseGroups=c("OSM","Carto"),overlayGroups=c("Chicago"))
  })
  output$table <- render_gt(expr = gt_tif_2022)
}
```

Running the Shiny App

```
shinyApp(ui = ui, server = server)
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.