```r
# ------------------------------------Documentation ----------------------------------
#
# By:       Hannah Fresques, ProPublica
# Date:     March 25, 2019
# Project:  Gutting the IRS
# Purpose:  Translate excel data to JSON.
#           Used for an interactive map (https://projects.propublica.org/graphics/eitc-audit)
#_____________________________________________________________________________________


# setup ------------------------------------------------------------------

library(readxl)
library(readr)
library(dplyr)
library(janitor)
library(purrr)
library(jsonlite)
library(stringr)

# read in data -----------------------------------------------------------

# estimated exams
counties <- read_xlsx(
  path="data/raw/Bloomquist - Regional Bias in IRS Audit Selection Data.xlsx",
  sheet="estimatedExams",
  col_types="text"
)


# filings
years <- 2012:2015

read_filings <- function(year){
  df <- read_xlsx(
    path=paste0("data/raw/County-",year,".xlsx"),
    skip=6,
    col_names=FALSE,
    col_types="text"
  )
  df <- df[,c(1:5)]
  colnames(df) <- c("State_FIPS_code","State","County_FIPS_code","County_name","Number_of_returns")
  df <- df %>%
    mutate(
      year=year
    )
}

filings <- years %>% map(read_filings) %>% bind_rows()
```

```r
# clean up filings data ----------------------------------------------------

filings %>% filter(is.na(County_FIPS_code)) %>% print(n=Inf)
# these are all notes from the bottom of files.
# drop them.

filings %>% filter(County_FIPS_code=="0") %>% count(County_name) %>% print(n=Inf)
# Except for DC, these are all state and country-wide totals.
# drop them.

filings2 <- filings %>%
  mutate(
    County_FIPS_code=case_when(
      State=="DC"~"001",
      TRUE~County_FIPS_code
    )
  ) %>%
  filter(!is.na(County_FIPS_code) & County_FIPS_code!="0") %>%
  mutate(
    fips = paste0(
      str_pad(State_FIPS_code , width=2, pad="0", side="left"),
      str_pad(County_FIPS_code, width=3, pad="0", side="left")
    )
  )

# wade became kusilvak
# shannon became oglala

filings3 <- filings2 %>%
  mutate(
    County_name=case_when(
      # named LaSalle Parish some years, La Salle Parish others. Just standardizing.
      fips=="22059"~"La Salle Parish",
      # use new names.
      fips=="02270"~"Kusilvak Census Area",
      fips=="46113"~"Oglala County",
      TRUE~County_name
    ),
    fips=case_when(
      # needs a real fips
      State=="DC"~"11001",
      # use old fips codes (because that's what the javascript mapping library expects)
      fips=="02158"~"02270",
      fips=="46102"~"46113",
      TRUE~fips
    )
  ) %>%
  group_by(fips,State,County_name) %>%
  summarize(
    years=n(),
    Number_of_returns = sum(as.numeric(Number_of_returns))
  )


# clean up counties data ---------------------------------------------------

counties <- counties %>%
```

```r
  clean_names() %>%
  mutate(
    # fips codes were missing leading zeros on the excel file
    fips = str_pad(fips, width=5, pad="0", side="left")
  )

counties %>% filter(fips %in% c("02270","46113","02158","46102"))
# this file uses the old fips codes and old county names for the SD and AK counties.

counties <- counties %>%
  mutate(
    county=case_when(
      # use new names.
      fips=="02270"~"Kusilvak Census Area",
      fips=="46113"~"Oglala County",
      TRUE~county
    )
    # keep old fips codes (because that's what the javascript mapping library expects)
  )


# put data together ------------------------------------------------------

counties2 <- counties %>%
  left_join(
    filings3,
    by="fips"
  )

counties2 %>% count(state,State) %>% print(n=Inf)


# check and clean data ---------------------------------------------------


counties3 <- counties2 %>%
  mutate(
    name = paste0(County_name,", ",state),
    estimated_exams = as.numeric(estimated_exams),
    Number_of_returns = as.numeric(Number_of_returns),
    audit_rate = (estimated_exams / Number_of_returns)*1000
  )


# national ---------------------------------------------------------------

national <- counties3 %>%
  summarize(
    estimated_exams=sum(estimated_exams),
    Number_of_returns=sum(Number_of_returns)
  )

# estimated_exams Number_of_returns
#         4506034         586148520

national_average <- (national$estimated_exams / national$Number_of_returns) * 1000
# national rate is 7.687529 per 1,000 filings
```

```r

# is anyone right on the average?
counties3 %>% filter(audit_rate==national_average) # none
counties3 %>% filter(audit_rate>national_average) %>% nrow() # 1514 above average
counties3 %>% filter(audit_rate<national_average) %>% nrow() # 1627 below average



# some checks ------------------------------------------------------------

library(ggplot2)
counties3 %>%
  ggplot(aes(x=audit_rate)) +
  geom_histogram()

# this distribution will not map well on a linear color scale of 0-12 per 1,000 filings.
# make a new value of audit_rate that has a floor and ceiling
counties3 <- counties3 %>%
  mutate(audit_rate_trunk=case_when(
    audit_rate<= 6 ~ 6,
    audit_rate>= 11 ~ 11,
    TRUE ~ audit_rate
  ))

counties3 %>%
  ggplot(aes(x=audit_rate_trunk)) +
  geom_histogram() +
  geom_vline(xintercept=national_average)




# save data --------------------------------------------------------------

counties4 <- counties3 %>%
  select(fips,name,state,Number_of_returns,estimated_exams,audit_rate,audit_rate_trunk)

# save to csv
# write_csv(counties4, "data/cleaned/auditsData_2019.04.03.csv")

# save to json
myJSON <- counties4 %>%
  transpose() %>%
  set_names(counties3$fips) %>%
  toJSON(auto_unbox = TRUE)
head(myJSON)
# save the text string as a json file
# fileConn<-file("data/cleaned/auditsData_2019.04.03.json")
# writeLines(myJSON, fileConn)
# close(fileConn)

```