Part 2: Web Scrapping

Step 0:

- library (knitr)
- opts_chunk$set(eval=FALSE)

Step 1: Assign the html as a string variable in R

- paa_url <- "http://paa2019.populationassociation.org/sessions/2"

Step 2: Assign a variable to read the html using the read_html() function

- paa_html <- read_html(paa_url)
- **paa_html**
    - **output**
        - {xml_document}
        - <html>
        - [1] <head>\n<meta http-equiv="Content-Type" content="text/html ...
        - [2] <body>\r\n<!--<div id="pnav"></div>-->\r\n\r\n<div class=" ...
        - ***This is telling us about the main "nodes" and where they are located***

Step 3: Look for the day and time of the session by inspecting the website.

Step 4: Extract date, time, location using the code below

- **Html_nodes () specifies the web page and the node you want to use**
- **Html_text () outputs the text of the specified node on the webpage**
- html_text((html_nodes(paa_html, "div.daytime")))
- ***or the below. They are the same***
- > paa_html %>%
- + html_nodes("div.daytime") %>%

- \+ html_text()
    - **Output**
        - [1] "\r\n \r\n Thursday, April 11 / \r\n 8:30 AM - 10:00 AM\r\n • \r\n Lone Star A (Level 3)\r\n \r\n"

Step 5: Clean up Step 4 and assign extracted text to a variable

- Day_time_place = html_text((html_nodes(paa_html, "div.daytime")), trim = TRUE)
- ***or the below. They are the same***
- day_time_place = paa_html %>%
- \+ html_nodes("div.daytime") %>%
- \+ html_text(trim=TRUE)
    - **Output**
        - [1] "Thursday, April 11 / \r\n 8:30 AM - 10:00 AM\r\n • \r\n Lone Star A (Level 3)"
- **By adding "trim = TRUE" it takes away the unnecessary spaces**
- **Below is tutorial's method using gsub**
    - day_time_place <- paa_html %>%
    - html_nodes("div.daytime") %>%
    - html_text() %>%
    - gsub("^\\s+|\\s+$", "", .)

Step 6: Find session number and session title

-

Step 7: Extract number and title and assign to variable

- Session_raw = html_text ((html_nodes (html_nodes (paa_html, "table"), "h2")))
- ***or the below. They are same***
- session_raw = paa_html %>%
- \+ html_nodes("table") %>%
- \+ html_nodes("h2") %>%

- \+ html_text()
    - **Output**
        - [1] "Session 2Flash Session: Population Aging, Consequences, and Public Policies"
- ***Here, we don't need to "trim" since it is more specific***

Step 8.1: Regex Tutorial

- Using Regex to match literal
- [\^$.|?*+() → have different meanings and are not interpreted literally like literals
    - You can specify a parameter an input has to match.
    - Output will be Bolean – TRUE or FALSE
        - Example 1: \$180
            - To have regex (or regular expression, %$#) in a literal sense precede it with \
            - = $180
        - Example 2: Lorem\sIpsum
            - \s matches white space
            - = Lorem Ipsum
        - Example 3: [013][FXB]
            - Anything containing 0,1,3 for first character and F,X,B for second character
            - 1X = True
        - Example 4: [1-4][A-Z]
            - Anything between 1 to 4 and between A to Z
            - 51 = False
            - Z1 = False
            - 1Z = True
            - ***It's order sensitive***
        - Example 5: [A-Za-z0-9]
            - Set multiple ranges for one character
            - The first character can be A to Z or a to z or 0 to 9
            - ***It's case sensitive***
        - Example 6: [^AEIOU]
            - Accepts everything except the parameter
        - Example 7: [-0-9]
            - Interprets the dash as literal if – is put in the beginning
            - As long as the number is between 0-9 it's True.
            - Number between 0-9 with a – in the front are also True
            - - → is also True
        - Example 8: ^[0-9]

        - Matches all things at the beginning of the line if it has a value from 0 to 9
        - ^[blah] → the ^ makes it so that you match the beginning of line
- Example 9: [0-9]$
    - Same as example 8 but end of line
- Example 10: ^[0-9][0-9]$
    - Specifying the start and end of the line
    - 1432 = False
    - **<u>This only accepts two character inputs for some reason</u>**
    - Example 10.1
        - [0,9][0,9]
        - 1432 = True
        - The parameters repeat in this case so as long as the number has an even amount of number it'll be true
    - **Explanation for example 10**
        - Reason 1432 is false is because the start as to be 1-9 and end has to be 1-9 and can only be 2 digits
        - If it were written as ^[0-9][0-9]…
            - It'll match the first two digits and display the rest of the line
            - But example ten, it won't display line until is satisfies all conditions
- Example 11: [A-Z]{3}
    - Repeats the range three times
    - Meaning a group of 3 characters with that range will count as one (not actually one, but like one correct entry)
    - YCA= True
- Example 12.1: [0-9]{3}[0-9]{3}[0-9]{4}
    - 10 digit phone number
    - 1231231231 = True
- Example 12.2: [0-9]{3}-[0-9]{3}-[0-9]{4}
    - Same but has to have dashes
- Example 13: [A-Z]{2,3}
    - Minimum as 2 repitition and max 3.
    - ***Can specific only min or max or both***
- Example 14: [0-9]?[A-Z]{2}
    - The question mark makes the part before it optional
    - AZ = True
    - 1AZ = True
    - Example 14.1: [0-9]{3}-?[0-9]{3}-?[0-9]{4}
        - This makes the dashes optional
- Example 15: [XYZ]+[1-9]+

        - The + is equivalent to {1,} meaning 1 minimum repetition
        - In this case any series of XYZ and 1-9 with however many reputations are grouped/matched
- Example 16: [0-3]+[XYZ]*
    - * is {0,} meaning that it completely optional
    - Difference with ? and * is that * will take as many repetitions
- **Clarification**
    - **If {x} only has single digit and not comma, it means that it must have exactly 3**
    - **If {,x} or {x,} then it's min or max**
    - **+ is {1,} * is {0,}**
    - **- is just a dash**
- Example 17: Dot **.**
    - Is a wildcard that represents literally anything even whitespace
- **My example: ^[1-9].*[A-Z]$**
    - **1asdasdsadasdaZ = TRUE**
- Example 18: ([A-Z][0-9]{3})+
    - Grouping
- Example 19: [0-9]{3}(35|75)
    - The | makes both inputs viable options. It represents "or."
        - 12375 = True
    - You have to enclose the parenthesis because if you don't…
        - [0-9]{3}35|75
            - The terms before | and after are grouped
            - 11135 = True
            - But Also
            - 75 = True
    - Example 19.1 ALPHA|BETA|GAMMA
        - Can use | more than once
- Example 20: (?<=regex) and (?=regex) [prefix and suffix]
    - Both matches whatever you replace with regex but does not include it
    - 1. Matches beginning
    - 2. Matches end
    - (?<=[A-Z]+)[0-9]+
        - ALPHA12
        - = 12
    - [0-9]+(?=[A-Z]+)
        - 12321312AZAZAZAZAZ
        - = 12321312

Step 8.2: Separate session number and title (when there are no obvious delimiters)

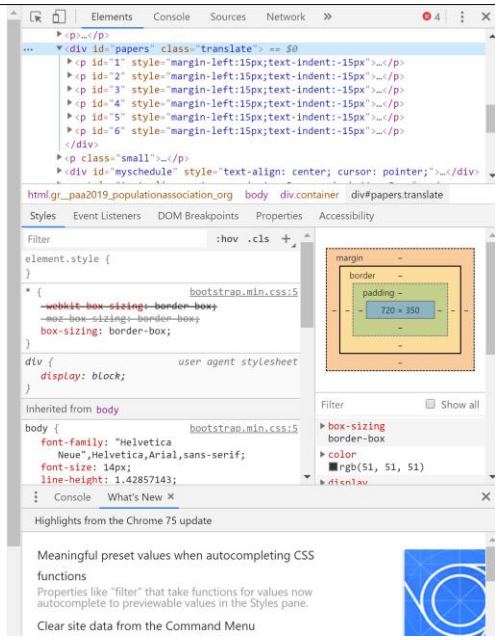
- Regexpr(x,y)
  - X is the parameter you define
  - Y is the source of where to apply the parameters
- Regmatches (x,y)
  - This defines what you want to output
  - X is the source
  - Y is the parameter you want to match
- **R Studio is a little limited in that if you want to remove items from the regexpr output you have to use gsub (x,y,z). Can't do all at once. But the logic for setting parameters are same.**
  - **Gsub (x,y,z)**
    - **X = what to exclude**
    - **Y = replace x with this**
    - **Z= Source**
- **Extracting session number**

```
> session_raw
[1] "Session 2Flash Session:  Population Aging, Consequences, and Public Policies"
> session_no_index=regmatches(session_raw, regexpr("^Session ([0-9]{1,3})", session_raw))
> session_no = gsub("[A-Za-z]+\\s", "", session_no_index)
> session_no
[1] "2"
```

- **Extracting title**

```
- > session_title = gsub("[A-Za-z]+\\s[1-9]+", "", session_raw)
- > session_title
- [1] "Flash Session:  Population Aging, Consequences, and Public Policies"
```

Step 9: Extract paper, author and institution



-
-

```
- > paa_html %>%
- +   html_nodes("div#papers") %>%
- +   html_text()
- [1] "\r\n      \r\n      1.\r\n      \r\n      Productive Aging in Developing Southeast Asia: Comparative Analyses Between Myanmar, Vietnam, and Thailand • \r\n\r\n      Bussarawan Teerawichitchainan , National University of SIngapore; Vipan Prachuabmoh, Chulalongkorn University; John Knodel, University of Michigan.      \r\n      \r\n      2.\r\n      \r\n      Living Alone in the United States and Europe: The Impact of Public Support on the Independence of Older Adults • \r\n\r\n      Stipica Mudrazija , Urban Institute; Jacqueline Angel, University of Texas at Austin; Ivan Cipin, University of Zagreb; Sime Smolic, University of Zagreb.      \r\n      \r\n      3.\r\n      \r\n      Long-Term Effects of a Conditional Cash Transfer Program on Adult Mortality: Evidence After 20 Years of Progressa • \r\n\r\n      Emma Aguila, University of Southern California; William H. Dow, University of California, Berkeley; Susan Parker , University of Maryland.      \r\n      \r\n      4.\r\n      \r\n      Aging and Population Policy in Developing Economies: Welfare Implications Across Generations • \r\n\r\n      Tanyasorn Ekapirak ; Minchung Hsu, GRIPS; Pei-Ju Liao, National Taiwan University.      \r\n      \r\n      5.\r\n      \r\n      SNAP and Food Consumption: A Collective Household Approach With Homescan Data • \r\n\r\n      Xirong Lin , Boston College.      \r\n      \r\n      6.\r\n      \r\n      Frequency of Benefit Payments in Social Transfer Programs • \r\n\r\n      Emma Aguila , University of Southern California; Arie Kapteyn, University of Southern California; Francisco Perez-Arce, University of Southern California.      \r\n    "
```

- **Notice that the node is div#papers unlike our previous div.class**
  - **You’ll know the difference. Inspect tells you what to use**

Step 10: Separate each section using strsplit()

- The whole list of topics 1-6 and clumped into one.
    - **In the above we can see that they are separated by a large amount of space, then a number, then a period, then another large space.**
        - **So it would be a good idea to split them by those sequence.**

```
- > strsplit(papers_raw, "\\s+[0-9]+[.]\\s+")
-
- [[1]]
- [1] ""
- [2] "Productive Aging in Developing Southeast Asia: Comparative Analyses Between
  Myanmar, Vietnam, and Thailand • \r\n\r\n Bussarawan Teerawichitchainan ,
  National University of SIngapore; Vipan Prachuabmoh, Chulalongkorn
  University; John Knodel, University of Michigan."
- [3] "Living Alone in the United States and Europe: The Impact of Public Support on the
  Independence of Older Adults • \r\n\r\n Stipica Mudrazija , Urban Institute;
  Jacqueline Angel, University of Texas at Austin; Ivan Cipin, University of
  Zagreb; Sime Smolic, University of Zagreb."
- [4] "Long-Term Effects of a Conditional Cash Transfer Program on Adult Mortality:
  Evidence After 20 Years of Progressa • \r\n\r\n Emma Aguila, University of
  Southern California; William H. Dow, University of California, Berkeley;
  Susan Parker , University of Maryland."
- [5] "Aging and Population Policy in Developing Economies: Welfare Implications
  Across Generations • \r\n\r\n Tanyasorn Ekapirak ; Minchung Hsu,
  GRIPS; Pei-Ju Liao, National Taiwan University."
- [6] "SNAP and Food Consumption: A Collective Household Approach With Homescan
  Data • \r\n\r\n Xirong Lin , Boston College."
- [7] "Frequency of Benefit Payments in Social Transfer Programs • \r\n\r\n
  Emma Aguila , University of Southern California; Arie Kapteyn, University of
  Southern California; Francisco Perez-Arce, University of Southern California.
  \r\n "
```

Step 11: Clean up step 10

```
> all_papers_raw <- unlist(strsplit(papers_raw, "\\s+[0-9]+[.]\\s+"))[-1]

> all_papers_raw

[1] "Productive Aging in Developing Southeast Asia: Comparative Analyses Between Myanmar,
Vietnam, and Thailand • \r\n\r\n Bussarawan Teerawichitchainan , National University
of SIngapore; Vipan Prachuabmoh, Chulalongkorn University; John Knodel,
University of Michigan."
```

```
[2] "Living Alone in the United States and Europe: The Impact of Public Support on the
Independence of Older Adults • \r\n\r\n Stipica Mudrazija , Urban Institute;
Jacqueline Angel, University of Texas at Austin; Ivan Cipin, University of Zagreb;
Sime Smolic, University of Zagreb."
[3] "Long-Term Effects of a Conditional Cash Transfer Program on Adult Mortality: Evidence
After 20 Years of Progressa • \r\n\r\n Emma Aguila, University of Southern California;
William H. Dow, University of California, Berkeley; Susan Parker , University of
Maryland."
[4] "Aging and Population Policy in Developing Economies: Welfare Implications Across
Generations • \r\n\r\n Tanyasorn Ekapirak ; Minchung Hsu, GRIPS;
Pei-Ju Liao, National Taiwan University."
[5] "SNAP and Food Consumption: A Collective Household Approach With Homescan Data •
\r\n\r\n Xirong Lin , Boston College."
[6] "Frequency of Benefit Payments in Social Transfer Programs • \r\n\r\n Emma
Aguila , University of Southern California; Arie Kapteyn, University of Southern
California; Francisco Perez-Arce, University of Southern California. \r\n "
```

- Notice that in step 10, the list was 1 list with 7 sub-elements.
    - The **unlist(x)** makes it into 7 individual elements
        - X = the list number
            - So if x where 2 it would remove the 2nd one
        - Remove more than one with
            - [-2:-4]
                - Get rid of list 2 through 4
        - Can also add previously removed element with [+x]
- Notice how step 10 has one empty element
    - **[-1]** get rid of that

Step 12: Write looping code to separate each element by topic, author and institution to be used for each element

- Step 12.1: First separate the two biggest categories: Paper and Author + Institution
    - `split_by_dot <- unlist(strsplit(all_papers_raw[1], "•"))`
- Step 12.2: Make a variable for only Title considering the situation where there is no specific Paper and only authors and institutions
    - If that is the case, we would except our list to be a length on 1
        - If it is 1, we want the title to be saved as “NA”
    - **Tutorial Code**
        - `paper_title <- ifelse(length(split_by_dot)==1, NA, gsub("[[:space:]]*$", "", split_by_dot[1]))`
        - **Interpretation**
        - ifelse (condition, x, y)
            - if True = x → NA
            - if False = y → gsub ()
    - **My version**

        -
        - if (length (split_by_dot==1){
        - paper_title = NA
        - }else {
        - (paper_title = gsub("\\s*$","", split_by_dot[1])
        - }

- Step 12.3: Make variable for author and institution considering situation with no paper
    - First we want to split the authors and institutions from each other
    - **Tutorial Code**
    - split_by_semicolon = ifelse(length(split_by_dot)==1, strsplit(split_by_dot[1], ";"),
    - strsplit(split_by_dot[2], ";"))
    - split_by_semicolon <- unlist(split_by_semicolon)

- [1] " \r\n\r\n Bussarawan Teerawichitchainan , National University of SIngapore"
- [2] " Vipan Prachuabmoh, Chulalongkorn University"
- [3] " John Knodel, University of Michigan."

    -
    - **Interpretation**
        - When there is not title, the length is going to be one and the author and institutions are going to be in the first line
        - Otherwise it's in the second
        - They are separated by a semicolon, so that is what we split them by

- Step 12.4: Clean spaces in 12.3

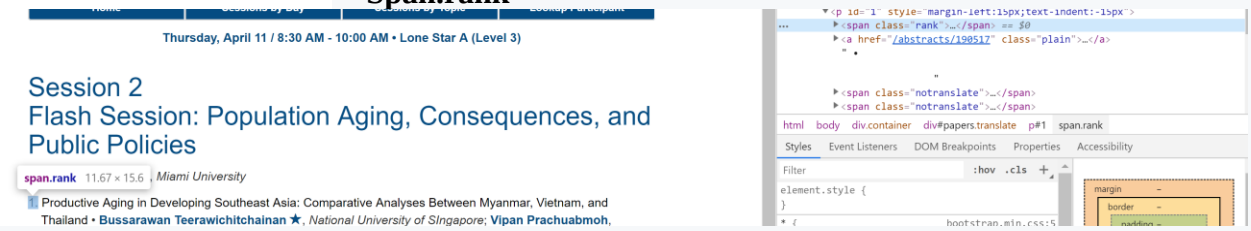
- > author_institution_vector <- gsub("^\\s+", "", split_by_semicolon)
- > author_institution_vector
- [1] "Bussarawan Teerawichitchainan , National University of SIngapore"
- [2] "Vipan Prachuabmoh, Chulalongkorn University"
- [3] "John Knodel, University of Michigan."

- Step 12.5: Combine them using cbind into a data frame

- > data.frame(cbind(paper_col, author_institution_vector))
- paper_col
- 1 Productive Aging in Developing Southeast Asia: Comparative Analyses Between Myanmar, Vietnam, and Thailand
- 2 Productive Aging in Developing Southeast Asia: Comparative Analyses Between Myanmar, Vietnam, and Thailand
- 3 Productive Aging in Developing Southeast Asia: Comparative Analyses Between Myanmar, Vietnam, and Thailand
- author_institution_vector
- 1 Bussarawan Teerawichitchainan , National University of SIngapore
- 2 Vipan Prachuabmoh, Chulalongkorn University
- 3 John Knodel, University of Michigan.

Step 13: Combine all the commands we created to create Loop

- We need to know how many times to iterate the loop
  - This would be the number of talks in the session which is 6.
    - The website counts this for us in…
      - **Span.rank**




```
> paa_html %>%
+   html_nodes("div#papers") %>%
+   html_nodes("span.rank")
{xml_nodeset (6)}
[1] <span class="rank">1<text>.</text></span>
[2] <span class="rank">2<text>.</text></span>
[3] <span class="rank">3<text>.</text></span>
[4] <span class="rank">4<text>.</text></span>
[5] <span class="rank">5<text>.</text></span>
[6] <span class="rank">6<text>.</text></span>
```

- Here we see that there are in fact 6 elements. So we can count the entries to use in iteration.

```
npapers <- paa_html %>%
  html_nodes("div#papers") %>%
  html_nodes("span.rank") %>%
  length()
```

Found out how to save R Script Files to File I want…Below notes are a copy of my R Scripts "Loop Session 2," "Loop all Sessions Outter Loop," "Loop all Poster Sessions," "Renaming"

```
npapers

for (p in 1:npapers){

#  just iterating the loop 6 times.

# whatever codes we input below will be interated 6 times


 split_by_dot = unlist(strsplit(all_papers_raw[p], "•"))

 # p represents how many times the loops has taken place.

 # Here we are just splitting the Paper from the authors and institutions

 # all_papers_raw has already been split by sections and unlisted


 paper_title = ifelse(length(split_by_dot)==1, NA, gsub("[[:space:]]*$", "", split_by_dot[1])
```

```
  # Code for when there is no paper title
  # If there isn't one, name it NA, or else Paper Title is just the title with the excess spaces removed

  split_by_semicolon = ifelse(length(split_by_dot)==1, strsplit(split_by_dot[1], ";"),
                              strsplit(split_by_dot[2], ";"))
  # Code for splitting the authors and institutions from other authors and institutions
  # If there is no Paper title, split element 1 by semicolon, else do the split in element 2

  split_by_semicolon = unlist(split_by_semicolon)
  author_institution_vector = gsub("^\\s+", "", split_by_semicolon)
  # Unlist it so they are individual elements, then get rid of unncessary space and rename

  paper_col = rep(paper_title, length(author_institution_vector))
  # Repeting the paper title by the number of authors+institutions

  new_df = data.frame(cbind(paper_col, author_institution_vector))
  # Combine the repeated paper title with each author per section so each author can be represented by
Paper separately

  if(p == 1){
    session_df <- new_df
  } else {
    session_df <- rbind(session_df, new_df)
  # If this is the first loop new_df = session_df, otherwise session_df equals it's current data frame plus
the new data frame established by current loop
  }
}

session_df
```

```
#######################################
# Creating Loop across all Sessions
#######################################

paa_df <- data.frame()
# empty data frame that we're going to fill in as we go



nsessions <- 5 # ideally needs to be automated
# we would assign the nodes that represents the sessions



# Beginning of loop for all sessions
for(sess in 1:nsessions){
  # here we're making an outter loop to be executed across all sections, or in this case 5 sessions

  Sys.sleep (3)
  # makes the program wait 3 seconds before moving onto the next session

  paa_url <- paste("http://paa2019.populationassociation.org/sessions/", sess)
  # sess here is the iterating variable we defined in our for loop
  paa_html <- read_html(paa_url)
  # read the specified session

  # And do all the processes the same

  day_time_place <- paa_html %>%
    html_nodes("div.daytime") %>%
    html_text() %>%
```

```
  gsub("^\\s+|\\s+$", "", .)

session_raw <- paa_html %>%
  html_nodes("table") %>%
  html_nodes("h2") %>%
  html_text()



# Note to self: always name the initial extration somethingsomething_raw
# And don't overwrite variables to create new ones. You may need them later


session_no_index <- regmatches(session_raw, regexpr("^Session ([0-9]{1,3})", session_raw))
session_no <- gsub("[A-Za-z]+ ", "", session_no_index)


session_title <- gsub("Session [0-9]{1,3}", "", session_raw)


papers_raw <- paa_html %>% # raw form
  html_nodes("div#papers") %>%
  html_text()

all_papers_raw <- unlist(strsplit(papers_raw, "\\s+[0-9]+[.]\\s+"))[-1]


npapers <- paa_html %>%
  html_nodes("div#papers") %>%
```

```
  html_nodes("span.rank") %>%
  length()

 # Beginning of loop for each paper in each session
 for(p in 1:npapers){
  split_by_dot <- unlist(strsplit(all_papers_raw[p], "•")) # first element is paper name, second element
is author/institution
  paper_title <- ifelse(length(split_by_dot)==1, NA, gsub("[[:space:]]*$", "", split_by_dot[1]))

  split_by_semicolon <- ifelse(length(split_by_dot)==1, strsplit(split_by_dot[1], ";"),
                               strsplit(split_by_dot[2], ";"))
  split_by_semicolon <- unlist(split_by_semicolon)
  author_institution_vector <- gsub("^\\s+", "", split_by_semicolon)

  paper_col <- rep(paper_title, length(author_institution_vector))
  new_df <- data.frame(cbind(paper_col, author_institution_vector))

  if(p == 1){
    session_df <- new_df
    # If, it's the first loop, the new data frame created equals the session data frame

  } else {
    session_df <- rbind(session_df, new_df)
    # If not the first loop, then add the data frame from that loop to existing frame
  }
 }
 # End of loop for each paper
```

```
  # Continued loop for all sessions
  date_col <- rep(day_time_place, nrow(session_df))
  # this repeats the data, time and place by the amount of sections there's been for the current session



  session_no_col <- rep(session_no, nrow(session_df))
  # repeats the session number by the amount of sections in the session


  session_title_col <- rep(session_title, nrow(session_df))
  # same thing
  # Top 3 codes were written since all the section within the same session have the same data, session number, and title
  # This would represent the data like this...
    # Date  Sess#   SessTitle
    #   1     2       A
    #   1     2       A
    #   1     2       A


  session_df <- cbind(session_no_col, session_title_col, session_df, date_col)
  # combines all the vertexes to create data frame for the current session


  if(sess == 1){
    paa_df <- session_df
    # if this is the first loop for a session, then paa data frame equals the data frame we just made in this current loop
  } else {
    paa_df <- rbind(paa_df, session_df)
  }
}
```

```
  # if not, combine the data frame of current session to existing data frame
View(paa_df)


colnames(paa_df) <- c("session_no", "session_title", "project_name", "author_institution",
"date_place")
colnames(ps_df) <- colnames(paa_df)



sessions <- rbind(paa_df, ps_df)


View(sessions)
```

```
############################
# Loop all POSTER SESSIONS
############################

# No explanation since, it's the same concept

ps_df <- data.frame()

npsessions <- 3 # ideally needs to be automated

for(psess in 1:npsessions){

 # IMPORTANT
 stop('Dont be scum, respect the web site. Intentionally build in wait time, use ?Sys.sleep')

 # Comment out the `stop()` line above
 # AND
 # Fill in Sys.sleep() Code Here



 ps_url <- paste0("http://paa2019.populationassociation.org/sessions/P", psess)
 ps_html <- read_html(ps_url)

 ### Day/time/place raw column (will be split later)
 day_time_place <- ps_html %>%
  html_nodes("div.daytime") %>%
  html_text() %>%
  gsub("^\\s+|\\s+$", "", .)
```

```
### Session number & session title:
session_raw <- ps_html %>%
  html_nodes("table") %>%
  html_nodes("h2") %>%
  html_text()

session_no_index <- regmatches(session_raw, regexpr("^Poster Session\\s ([0-9]+)", session_raw))
session_no <- gsub("[A-Za-z ]+ ", "", session_no_index)

session_title <- gsub("Poster Session\\s [0-9]+", "", session_raw)

### Poster name/author name/institution name:
papers_raw <- ps_html %>% # raw form
  html_nodes("div#papers") %>%
  html_text()

all_papers_raw <- unlist(strsplit(papers_raw, "\\s+[0-9]{1,3}[.]\\s+"))[-1]

# session_df <- data.frame()

# total number of papers for a session:
nposters <- ps_html %>%
  html_nodes("div#papers") %>%
  html_nodes("span.rank") %>%
  length()

for(p in 1:nposters){
```

```
  split_by_dot <- unlist(strsplit(all_papers_raw[p], "•")) # first element is paper name, second element is author/institution
  paper_title <- ifelse(length(split_by_dot)==1, NA, gsub("[[:space:]]*$", "", split_by_dot[1]))

  split_by_semicolon <- ifelse(length(split_by_dot)==1, strsplit(split_by_dot[1], ";"),
                               strsplit(split_by_dot[2], ";"))
  split_by_semicolon <- unlist(split_by_semicolon)
  author_institution_vector <- gsub("^\\s+", "", split_by_semicolon)

  paper_col <- rep(paper_title, length(author_institution_vector))
  new_df <- data.frame(cbind(paper_col, author_institution_vector))

  if(p == 1){
    session_df <- new_df
  } else {
    session_df <- rbind(session_df, new_df)
  }
}

date_col <- rep(day_time_place, nrow(session_df))
session_no_col <- rep(session_no, nrow(session_df))
session_title_col <- rep(session_title, nrow(session_df))

session_df <- cbind(session_no_col, session_title_col, session_df, date_col)

if(psess == 1){
  ps_df <- session_df
} else {
  ps_df <- rbind(ps_df, session_df)
```

```
  }

}

View(ps_df)
```

```
# Making the column names for Poster and Sessions so we can combine

colnames(paa_df) <- c("session_no", "session_title", "project_name", "author_institution",
"date_place")

colnames(ps_df) <- colnames(paa_df)


sessions <- rbind(paa_df, ps_df)


View(sessions)



# Poster and Sessions share the same numbers 1-11, so the column representing session number may
appear twice without any indication of which it is

session$poster = c(rep(0, nrow(paa_df)), rep(1, nrow(ps_df)))

# this creates a vector that assigns a binary of 0 for each entry of normal sessions and 1 for posters

# What is $poster? Is it just a name?


# Relabeling the column names so they aren't factors

sessions$project_name <- as.character(sessions$project_name)

sessions$author_institution <- as.character(sessions$author_institution)

sessions$date_place <- as.character(sessions$date_place)


# Extra practice: split the author and institutions

# Same thing as before

regexpr(",", sessions$author_institution)

# This "," is basically saying you only want to see the ","


regmatches(sessions$author_institution,

        regexpr(",", sessions$author_institution), invert = TRUE)
```

```
# Because of the invert = TRUE, the comma is the only thing that'll be removed

split_author_inst <- regmatches(sessions$author_institution,
                                regexpr(",", sessions$author_institution), invert = TRUE)

unlist(lapply(split_author_inst, `[[`, 1))

sessions$author <- unlist(lapply(split_author_inst, `[[`, 1))
```