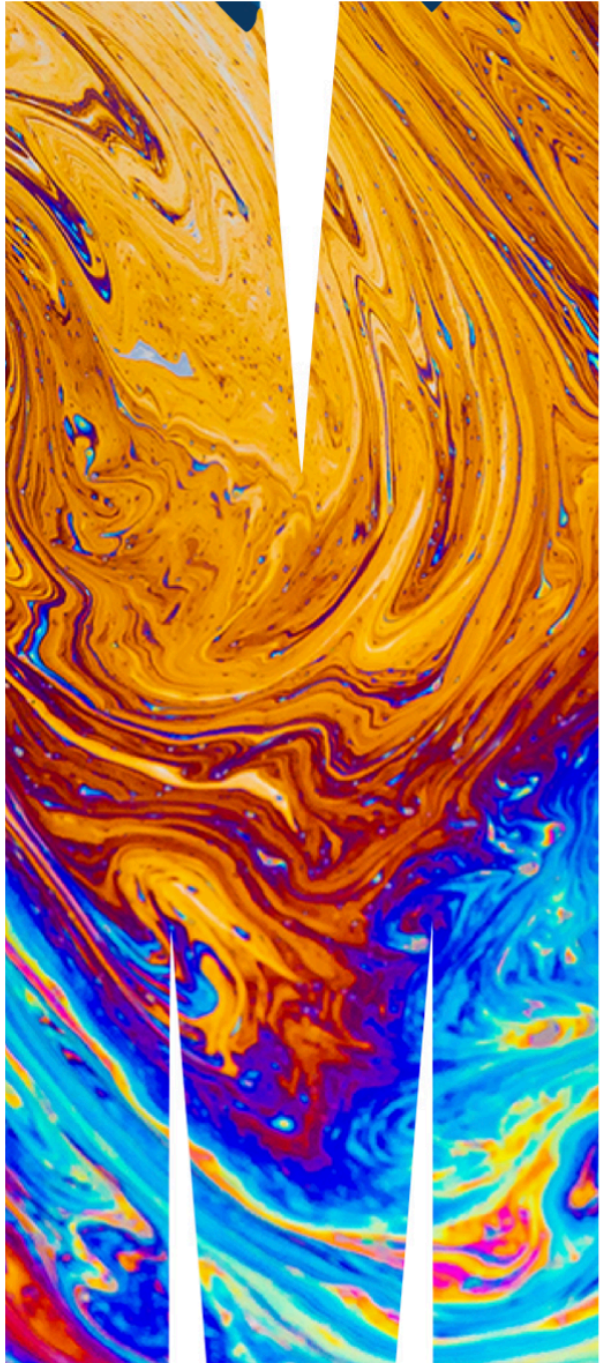
MONASH University

ETC5512: Wild Caught Data

Introduction to web scraping



Lecturer: *Kate Saunders*

Department of Econometrics and Business Statistics

ETC5512.Clayton-x@monash.edu

Week 12

Motivation

We are R-Ladies
@WeAreRLadies · **Follow**

The most important thing I've ever done for learning R is to paradoxically stop "learning" (e.g. classes and problem sets) and start doing. Take a problem you have at work or school or a dataset you find interesting and get to work. Then write it up and post on githib or a blog.

Matt Motyl @MattMotyl

Replying to @WeAreRLadies

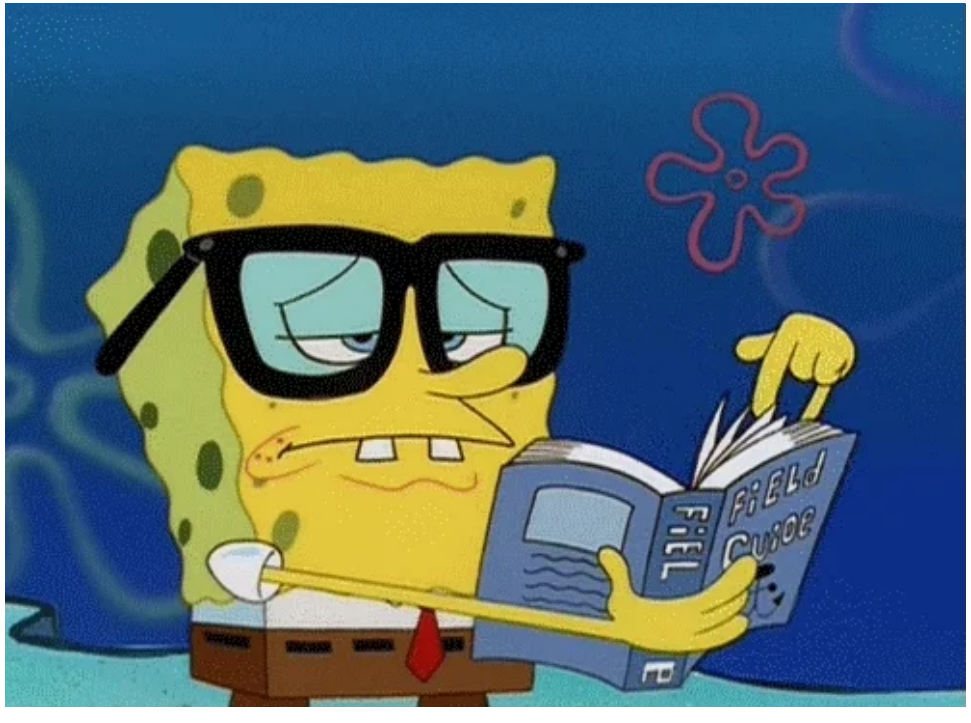
What did you find most helpful in learning/studying R? I've come along way, but have a long way to go and am blown away by some of the things I see from @WeAreRLadies. Any recs / tips are greatly appreciated.

4:19 AM · Mar 26, 2019

478 Reply Copy link

Read 8 replies

After PhD - I wanted to read more!

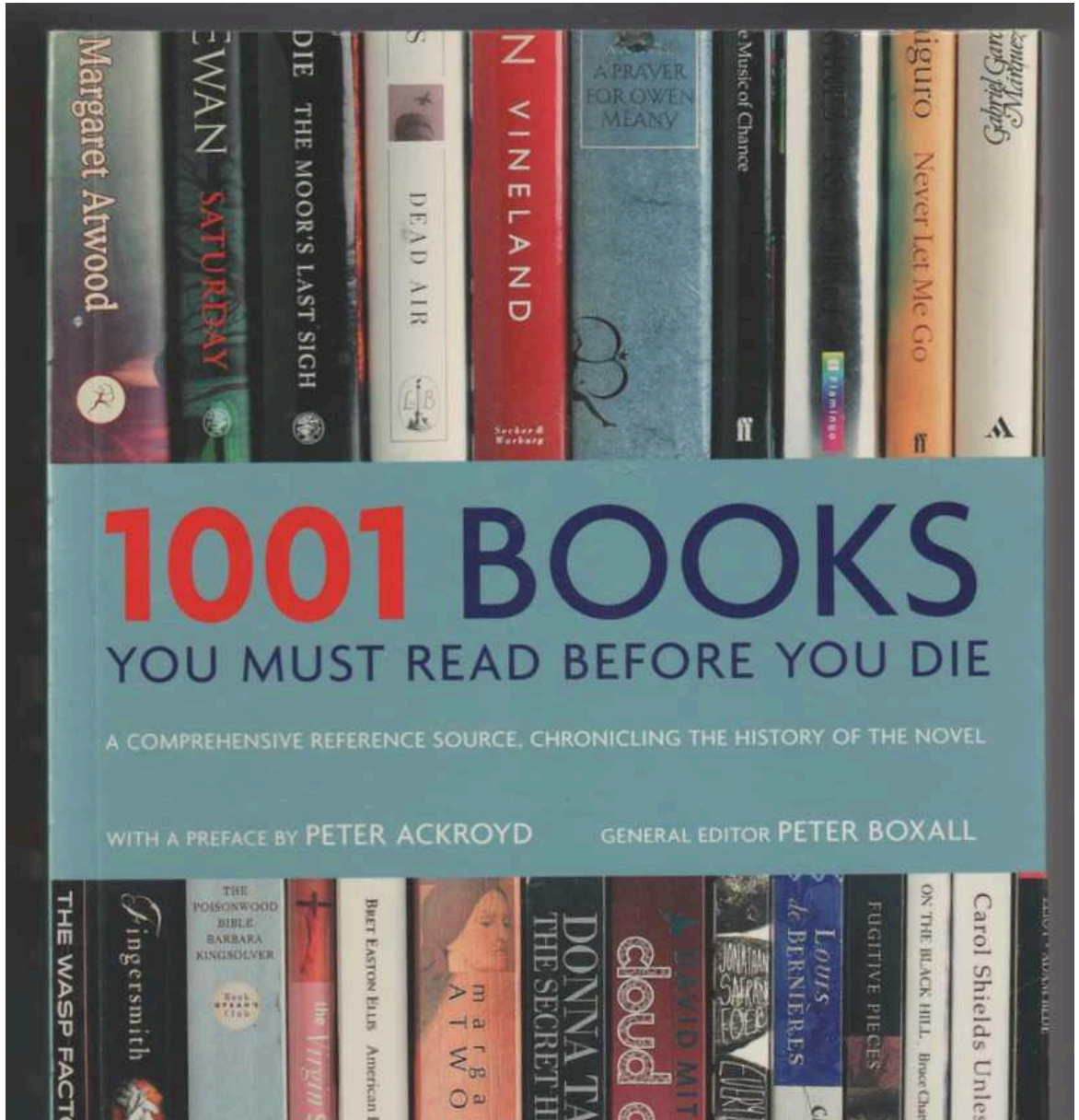


Question

How should I choose what to read?

Am I reading diversely?

Picking from lists



Questions for our data

Are lists like this biased?

Today you will:

- Get to know the basics of webpages
- Look at some examples of webscraping
- Get some data to answer my questions

Coding Perspective:

- Learn how to read data from a webpage into R
- Do more string manipulation
- Learn about automating a scraper

Packages we need

`rvest` is the R package that we'll need to get started with learning the 101 of web scraping

```
library(rvest)
library(tidyverse)
```

- Good for static webpages

Webpage basics

Go to a webpage

https://en.wikipedia.org/wiki/Tim_Winton

Article Talk Read Edit View history Search Wikipedia

Tim Winton

From Wikipedia, the free encyclopedia

Timothy John Winton (born 4 August 1960) is an Australian writer of novels, children's books, non-fiction books, and short stories. In 1997 he was named a Living Treasure by the National Trust, and has won the Miles Franklin Award four times.

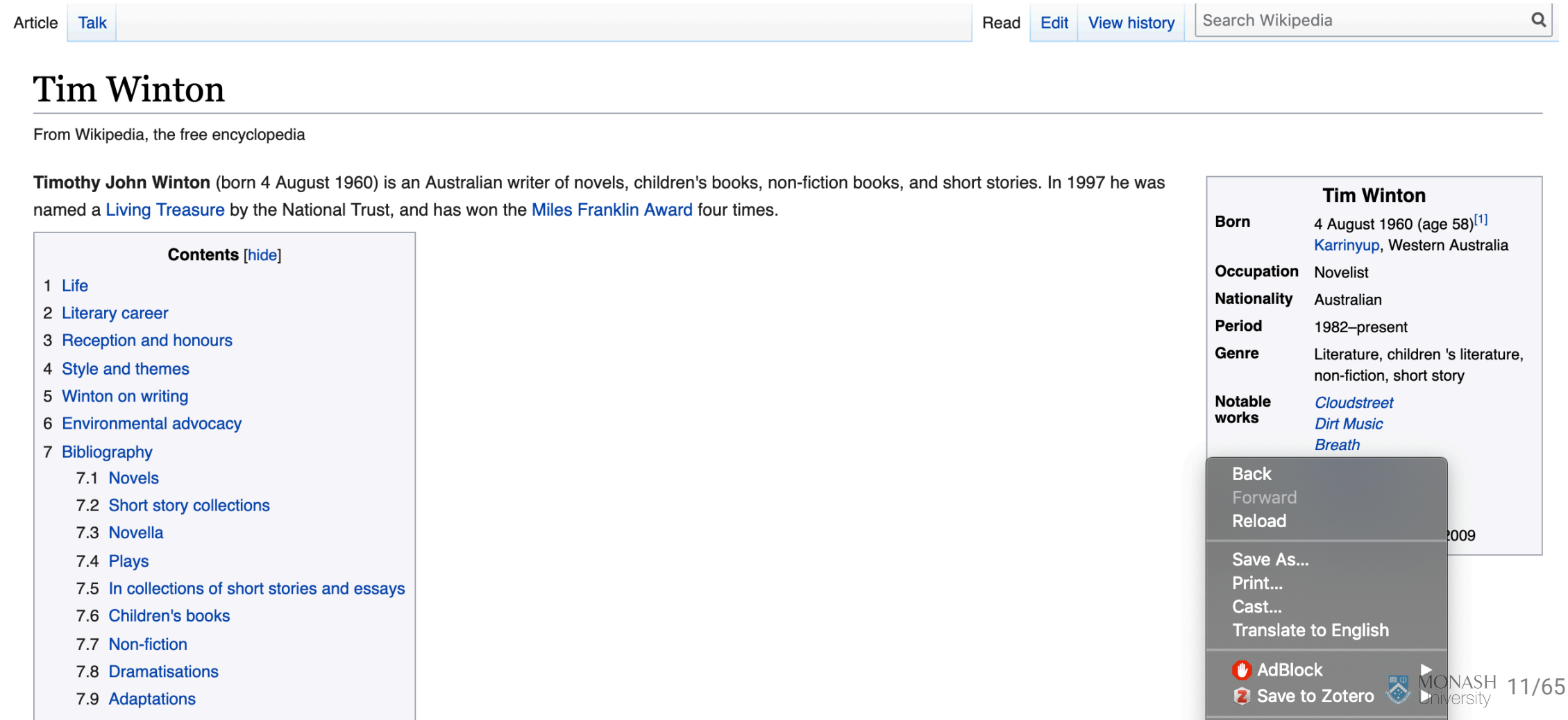
Contents [hide]

Tim Winton

Born	4 August 1960 (age 58)[1] Karrinyup, Western Australia
Occupation	Novelist
Nationality	Australian
Period	1982–present
Genre	Literature, children 's literature, non-fiction, short story
Notable works	*Cloudstreet* *Dirt Music* *Breath* *Shallows* *Blueback*
Notable awards	Miles Franklin 1984, 1992, 2002, 2009

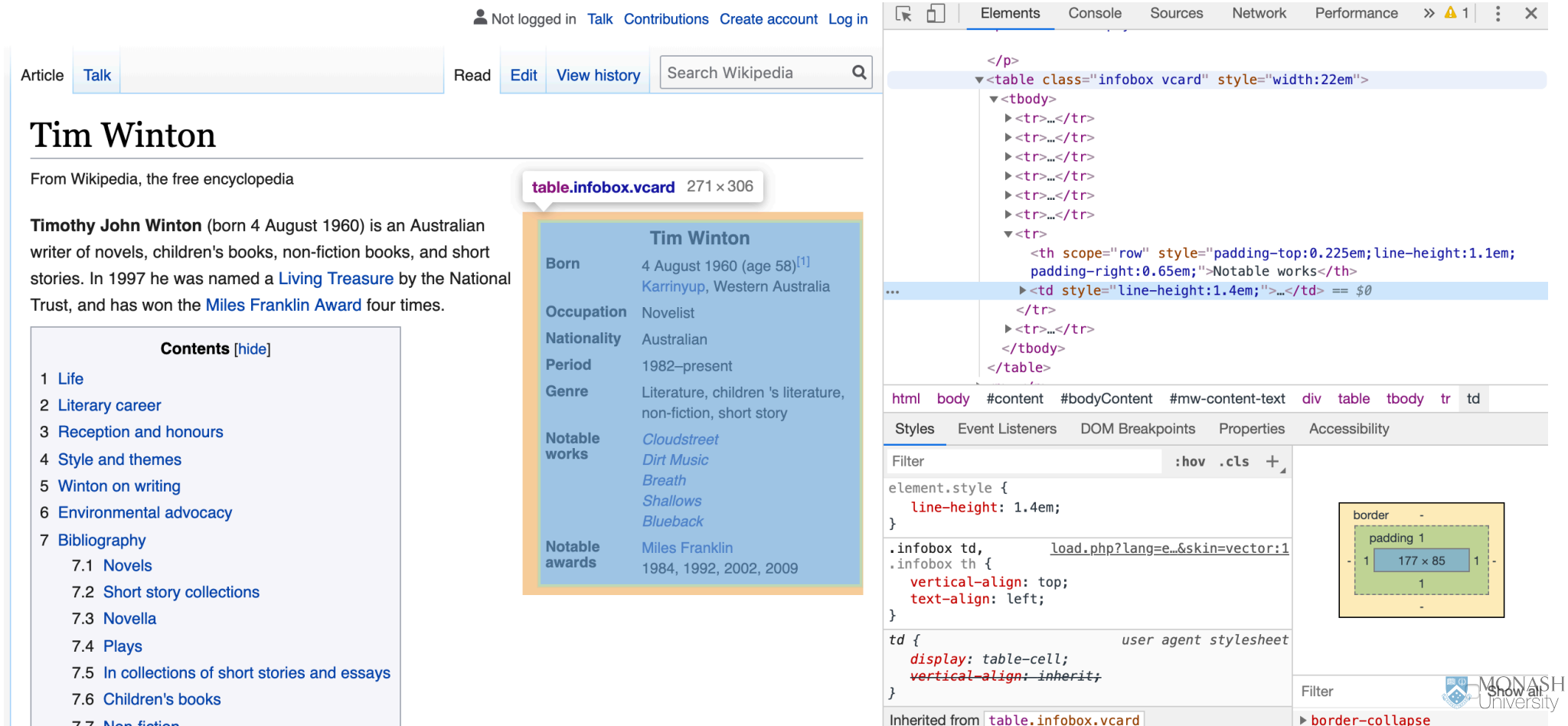
View html code in Chrome

- Right click the part of the page you want
- Select inpsect



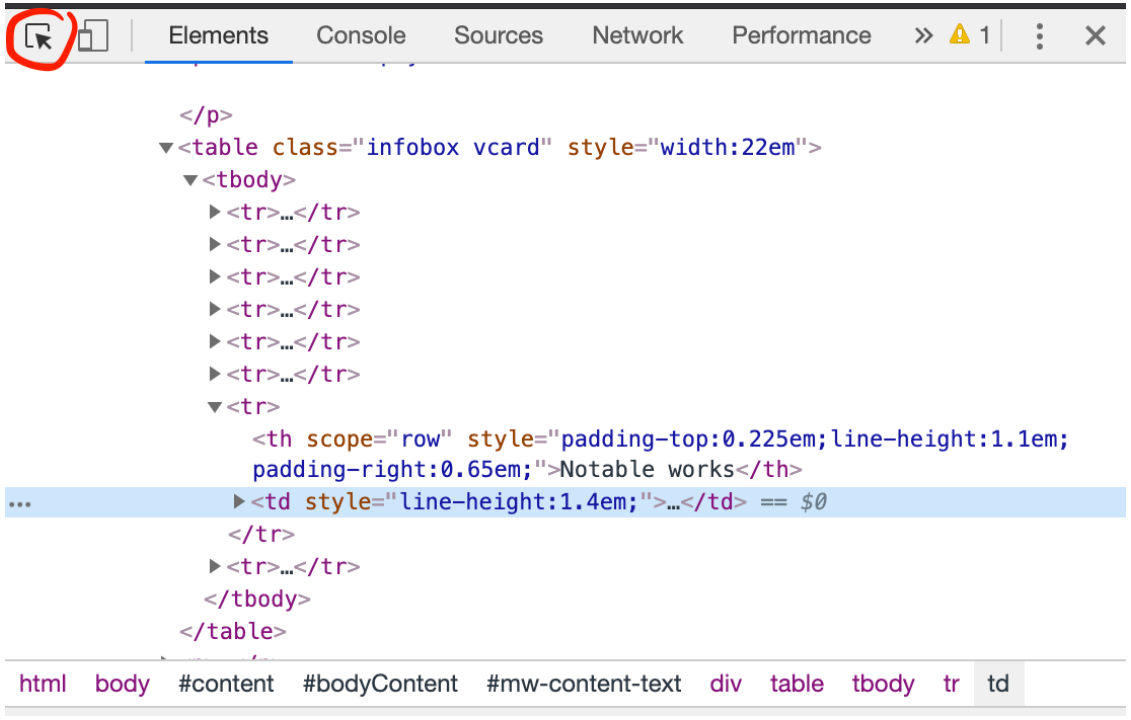
Html code

- Brings up the html code
- Highlights the piece of html code related to your click
- Hover over html code to see other features of the web page



Inpsect button

- Similarly, click the top left button in the side panel
- Explore related features of the webpage and html code



Basic html types

By browsing you observe the basic **structure** of html webpages

Opening and closing tags wrapped around content to define its purpose and appearance on a webpage.

e.g. < tag > lorem ipsem text < /tag >

Some basic **tag types** are:

div - Division or section

table - Table

p - Paragraph elements

h - Heading

Read a webpage

```
library(rvest)
author_url <- "https://en.wikipedia.org/wiki/Tim_Winton"
wiki_data <- read_html(author_url) # Read the webpage into R

str(wiki_data)

## List of 2
##  $ node:<externalptr>
##  $ doc :<externalptr>
##  - attr(*, "class")= chr [1:2] "xml_document" "xml_node"

wiki_data

## {html_document}
## <html class="client-nojs vector-feature-language-in-header-enabled vector-fea
## [1] <head>\n<meta http-equiv="Content-Type" content="text/html; charset=UTF-8
## [2] <body class="skin-vector skin-vector-search-vue mediawiki ltr sitedir-ltr
```

How to scrape a table - html_table()

So we can read data from the website into R, but we need the data in a form we can use.

```
table_data <- wiki_data |>
  rvest::html_table(header = FALSE) #Get all tables on the webpage

length(table_data)

## [1] 5

table_data[[1]]

## # A tibble: 9 × 2
##   X1                                                  X2
##   <chr>                                               <chr>
## 1 Tim WintonAO                                        Tim WintonAO
## 2 Winton at the launch of Breath in London, 2008      Winton at the launch of Brea
## 3 Born                                                Timothy John Winton4 August
## 4 Occupation                                          Novelist
## 5 Nationality                                         Australian
## 6 Period                                              1982–present
```

Other approaches - html_nodes()

```
table_data_eg1 <- wiki_data |>
 rvest::html_nodes("table") |> # get all the nodes of type table
 purrr::pluck(1) |> #pull out the first one
 rvest::html_table(header = FALSE) #convert it to table type

table_data_eg1
```

```
## # A tibble: 9 × 2
##   X1                                                X2
##   <chr>                                             <chr>
## 1 Tim WintonAO                                      Tim WintonAO
## 2 Winton at the launch of Breath in London, 2008 Winton at the launch of Brea
## 3 Born                                              Timothy John Winton4 August
## 4 Occupation                                        Novelist
## 5 Nationality                                       Australian
## 6 Period                                            1982–present
## 7 Genre                                             Literature, children's, non-
## 8 Notable works                                     Cloudstreet Dirt Music Breat
```

Other approaches - html_node()

Lots of functions in `rvest` give you the option to return the first match or to return all matches.

```
table_data_eg2 <- wiki_data |>
  rvest::html_node("table") |> # just get the first table match
  rvest::html_table(header = FALSE) #convert it to table type

table_data_eg2
```

```
## # A tibble: 9 × 2
##   X1                                                  X2
##   <chr>                                               <chr>
## 1 Tim WintonAO                                        Tim WintonAO
## 2 Winton at the launch of Breath in London, 2008      Winton at the launch of Brea
## 3 Born                                                Timothy John Winton4 August
## 4 Occupation                                          Novelist
## 5 Nationality                                         Australian
## 6 Period                                              1982–present
## 7 Genre                                               Literature, children's, non-
## 8 Notable works                                       Cloudstreet Dirt Music Breat
```

Get the Nationality

One more step - Need to get the nationality from the table

```
author_nationality = table_data_eg2 |>
  dplyr::rename(Category = X1, Response = X2) |>
  dplyr::filter(Category == "Nationality") |>
  dplyr::select(Response) |>
  as.character()

author_nationality

## [1] "Australian"
```

Now the real challenge: **Can we generalise?**

Breakout Session

Try it yourself time:

- Pick your an author and find their wikipedia page
- Explore the structure of the webpage
- Download their infocard into R
- Can you get their nationality from the infocard?

Let's try a different author

"https://en.wikipedia.org/wiki/Jane_Austen"

Jane Austen (/ˈɒstɪn, ˈɔːs-/; 16 December 1775 – 18 July 1817) was an English novelist known primarily for her six major novels, which interpret, critique and comment upon the British landed gentry at the end of the 18th century. Austen's plots often explore the dependence of women on marriage in the pursuit of favourable social standing and economic security. Her works critique the novels of sensibility of the second half of the 18th century and are part of the transition to 19th-century literary realism.[2][b] Her use of biting irony, along with her realism, humour, and social commentary, have long earned her acclaim among critics, scholars, and popular audiences alike.[4]

With the publications of *Sense and Sensibility* (1811), *Pride and Prejudice* (1813), *Mansfield Park* (1814) and *Emma* (1816), she achieved success as a published writer. She wrote two additional novels, *Northanger Abbey* and *Persuasion*, both published posthumously in 1818, and began another, eventually titled *Sanditon*, but died before its completion. She also left behind three volumes of juvenile writings in manuscript, a short epistolary novel *Lady Susan*, and another unfinished novel, *The Watsons*. Her six full-length novels have rarely been out of print, although they were published anonymously and brought her moderate success and little fame during her lifetime.

A significant transition in her posthumous reputation occurred in 1833, when her novels were republished in Richard Bentley's Standard Novels series, illustrated by Ferdinand Pickering, and sold as a set.[5] They gradually gained wider acclaim and popular readership. In 1869, fifty-two years after her death, her nephew's publication of *A Memoir of Jane Austen* introduced a compelling version of her writing career and supposedly uneventful life to an eager audience.

Austen has inspired a large number of critical essays and literary anthologies. Her novels have inspired many films, from 1940's *Pride and Prejudice* to more recent productions like *Sense and Sensibility* (1995), *Emma* (1996), *Mansfield Park* (1999), *Pride & Prejudice* (2005), and *Love & Friendship* (2016).

Contents [hide]

Jane Austen

Portrait, c. 1810[a]

Born	16 December 1775 Steventon Rectory, Hampshire, England
Died	18 July 1817 (aged 41) Winchester, Hampshire, England
Resting place	Winchester Cathedral, Hampshire, England
Education	Reading Abbey Girls' School
Period	1787 to 1809–11
Relatives	James Austen (brother) George Austen (brother) Edward Austen Knight (brother) Henry Thomas Austen (brother) Cassandra Austen (sister) Sir Francis Austen (aka Francis,

Generalise the web page

```
author_first_name = "Jane"
author_last_name = "Austen"
author_url <- paste("https://en.wikipedia.org/wiki/",
  author_first_name, "_", author_last_name, sep = "")
wiki_data <- read_html(author_url)
```

Let's get that table

```
table_data <- wiki_data |>
  rvest::html_nodes(".infobox.vcard") |> #search for a class
  rvest::html_table(header = FALSE) |>
  purrr::pluck(1)

head(table_data)
```

```
## # A tibble: 6 × 2
##   X1                    X2
##   <chr>                 <chr>
## 1 Jane Austen           Jane Austen
## 2 Portrait, c. 1810[a] Portrait, c. 1810[a]
## 3 Born                  (1775-12-16)16 December 1775Steventon Rectory, Hampshi
## 4 Died                  18 July 1817(1817-07-18) (aged 41)Winchester, Ham
## 5 Resting place         Winchester Cathedral, Hampshire
## 6 Period                1787–1817
```

Web scraping is tricky

```
table_data |> dplyr::select(1) |> unlist() |> as.vector()
```

```
## [1] "Jane Austen"          "Portrait, c. 1810[a]" "Born"
## [4] "Died"                 "Resting place"        "Period"
## [7] "Relatives"            "Signature"            ""
```

No nationality category in Jane Austen's infocard

Her nationality is in the webpage text.

Let's scrape the nationality from there.

Try another way

```
para_data <- wiki_data |>
  rvest::html_nodes("p") # get all the paragraphs
head(para_data)

## {xml_nodeset (6)}
## [1] <p class="mw-empty-elt">\n\n\n\n</p>
## [2] <p><b>Jane Austen</b> (<span class="rt-commentedText nowrap"><span class=
## [3] <p>The anonymously published <i><a href="/wiki/Sense_and_Sensibility" tit
## [4] <p>Since her death Austen's novels have rarely been out of print. A signi
## [5] <p>The scant biographical information about Austen comes from her few sur
## [6] <p>The first Austen biography was <a href="/wiki/Henry_Thomas_Austen" tit
```

But where exactly do we find her nationality in all this text?

Let's go back to exploring the webpage.

Get the text - html_text()

```
text_data <- para_data |>
  purrr::pluck(2) |> # get the second paragraph
  rvest::html_text() # convert the paragraph to text
head(text_data)

## [1] "Jane Austen (/ˈɒstɪn, ˈɔːstɪn/ OST-in, AW-stin; 16 December 1775 – 18
```

Let's look at two other ways we could do this.

Xpath Example

- Right click html code, copy, copy Xpath

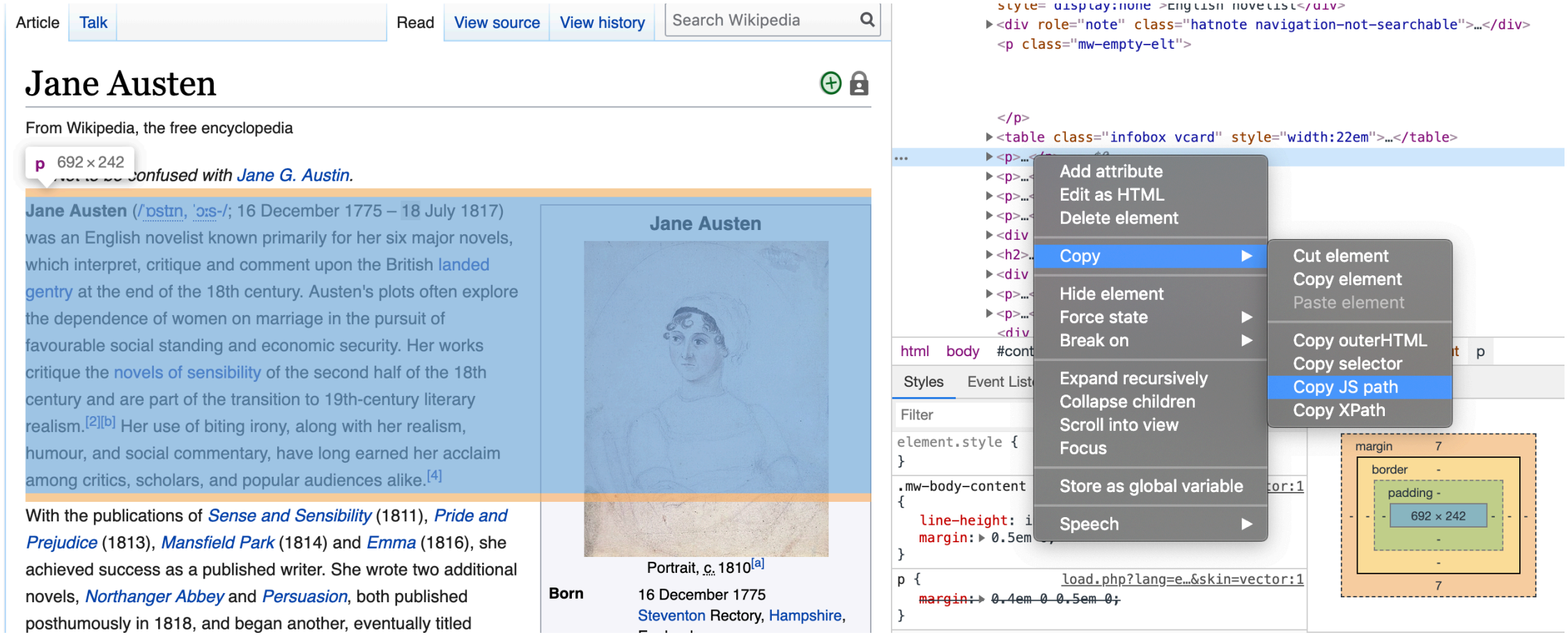
Using an Xpath

```
para_xpath = '//*[@id="mw-content-text"]/div/p[2]'
text_data <- wiki_data |>
  rvest::html_nodes(xpath = para_xpath) |>
  rvest::html_text()
text_data

## [1] "Jane Austen (/ˈɒstɪn, ˈɔːstɪn/ OST-in, AW-stin; 16 December 1775 – 18
```

JSpath Example

- Right click html code, copy, copy JS path



Using CSS ID

```
para_css = "#mw-content-text > div > p:nth-child(5)"
text_data <- wiki_data |>
  rvest::html_nodes(css = para_css) |>
  rvest::html_text()
text_data

## [1] "Jane Austen (/ˈɒstɪn, ˈɔːstɪn/ OST-in, AW-stin; 16 December 1775 – 18
```

Text Analysis

Still need to get her nationality, use `str_count`

```
possible_nationalities <- c("Australian", "Chinese", "Mexican", "English", "Ethi

# Do any of these nationalities appear in the text?
count_values = str_count(text_data, possible_nationalities)

count_values == TRUE # Which ones were matched

## [1] FALSE FALSE FALSE  TRUE FALSE

possible_nationalities[count_values == TRUE] #Get the matching nationalities

## [1] "English"
```

- What do you think of my solution?
- Any guesses why I didn't use `str_match`?

Learnt so far

- Know how to explore a web page with inspect
- Know some basics about how to get data

Also know:

- Can be hard to generalise
- Formats aren't always standard

Back to the original question

Need to get our list

1,001 Books to Read Before Yo

https://mizparker.wordpress.com/the-lists/1001-books-to-read-before-you-die/

21st Century:

1. Never Let Me Go – Kazuo Ishiguro
2. Saturday – Ian McEwan
3. On Beauty – Zadie Smith
4. Slow Man – J.M. Coetzee
5. Adjunct: An Undigest – Peter Manson
6. The Sea – John Banville
7. The Red Queen – Margaret Drabble
8. The Plot Against America – Philip Roth
9. The Master – Colm Toibin
10. Vanishing Point – David Markson
11. The Lambs of London – Peter Ackroyd
12. Dining on Stones – Iain Sinclair
13. Cloud Atlas – David Mitchell
14. Drop City – T. Coraghessan Boyle
15. The Colour – Rose Tremain
16. Thursbitch – Alan Garner
17. The Light of Day – Graham Swift
18. What I Loved – Siri Hustvedt
19. **The Curious Incident of the Dog in the Night-Time – Mark Haddon**
20. Islands – Dan Sleigh

Recently Reviewed

- A Visit From the Goon Squad
- The Rent Collector
- Lullaby
- Memories of My Melancholy Whores (Memoria de Mis Putas Tristes)
- Sexing the Cherry
- Sweatpants & Coffee
- Forged In Grace
- Less Than Zero
- Lucky
- Mermaids in the Basement
- Beloved
- Trainspotting
- The Help
- The Year of Living Biblically
- Water for Elephants
- Lady Chatterley's Lover
- Atlas Shrugged
- I Know Why the Caged Bird Sings
- Love in the Time of Cholera (El Amor en los Tiempos del Cólera)
- The World According to Garp

Most Popular Reviews

Read the book list from a website

```
book_list_url <- "https://mizparker.wordpress.com/the-lists/1001-books-to-read-b
paragraph_data <- read_html(book_list_url) |> # read the web page
  rvest::html_nodes("p") # get the paragraphs
paragraph_data[1:12]

## {xml_nodeset (12)}
##  [1] <p>This list has appeared in several places around the internet, and is
##  [2] <p>If you would like to download a spreadsheet of the list and keep trac
##  [3] <p><strong>21st Century:</strong></p>
##  [4] <p>1.  Never Let Me Go – Kazuo Ishiguro<br>\n2.  Saturday – Ian McEwan<b
##  [5] <p><strong>20th Century:</strong></p>
##  [6] <p>70. Timbuktu – Paul Auster<br>\n71. The Romantics – Pankaj Mishra<br>
##  [7] <p><strong>19th Century</strong></p>
##  [8] <p>786. Some Experiences of an Irish R.M. – Somerville and Ross<br>\n787
##  [9] <p><strong>18th Century</strong></p>
## [10] <p>943. Hyperion – Friedrich Hölderlin<br>\n944. The Nun – Denis Diderot
## [11] <p><strong>Pre-1700</strong></p>
## [12] <p>989. Oroonoko – Aphra Behn<br>\n990. The Princess of Clèves – Marie-M
```

Get the book list from the paragraphs

This list is in pieces, but the format seems mostly consistent

```
book_string <- paragraph_data |>
  purrr::pluck(4) |> # get the first part of the book list
  html_text(trim = TRUE) |> # convert it to text, remove excess white space
  str_replace_all("\n", "")

head(book_string)

## [1] "1.  Never Let Me Go – Kazuo Ishiguro2.  Saturday – Ian McEwan3.  On Beau
```

More string manipulations

Web scraping often means string handling

We want to split the string by any numbers followed by a full stop

Careful:

- don't want to split book titles with numbers, like Catch 22,
- don't want to split authors with full stops, like J.R.R Tolkien

Actually bit tricky!

Resources:

- Lecture 4
- stringr cheatsheet from RStudio
- generative AI is also great resource

Do some string handling

```
eg_string = "9. book - author 10. book - author"

str_view(eg_string, "[:digit:]") #Match by any digit

## [1] | <9>. book - author <1><0>. book - author

str_view(eg_string, "[:digit:]+") #Match by one or more digits

## [1] | <9>. book - author <10>. book - author

str_view(eg_string, "\\.") #Match by fullstop

## [1] | 9<.> book - author 10<.> book - author

str_view(eg_string, "[[:digit:]]+?\\.") #Match digits followed by a fullstop

## [1] | <9.> book - author <10.> book - author
```

Get the list as a dataframe

```
books_df <- book_string |>
  str_split("[[:digit:]]+?\\.") |>
  as.data.frame(stringsAsFactors = FALSE)

names(books_df) = "Book_Author"

books_df = books_df |>
  dplyr::filter(Book_Author != "") |> # remove any empty rows
  tidyr::separate(Book_Author, sep = "\\-", into = c("book", "author")) # splits
```

Result

```
##                       book          author
## 1        Never Let Me Go   Kazuo Ishiguro
## 2               Saturday       Ian McEwan
## 3             On Beauty      Zadie Smith
## 4              Slow Man     J.M. Coetzee
## 5   Adjunct: An Undigest     Peter Manson
## 6               The Sea    John Banville
```

We are very lucky!

- Easily split our author and book into columns
- Thanks to whoever coded this webpage using a long hash!

Need to repeat it: So wrap code in a function

```
Convert_book_string_to_df <- function(para_ind, pargraph_data){

  book_string <- paragraph_data |>
  purrr::pluck(para_ind) |>
  html_text(trim = TRUE) |>
  str_replace_all("\n", "")

  books_df <- book_string |>
    str_split("[[:digit:]]+?\\.") |>
    as.data.frame(stringsAsFactors = FALSE)

  names(books_df) = "Book_Author"

  books_df = books_df |>
    dplyr::filter(Book_Author != "") |>
    tidyr::separate(Book_Author, sep = "\\-", into = c("book", "author"))

  return(books_df)}
```

Get the final list

Need to run this function for every second paragraph starting from number 4 until paragraph 12.

```
book_data <- lapply(seq(4,12,2) %>% as.list(),
                    Convert_book_string_to_df, paragraph_data) %>%
  do.call(rbind, .) %>%
  dplyr::mutate(author = str_trim(author))

nrow(book_data) # Has 1001 rows

## [1] 1001
```

Check what it looks like

Randomly pick 10 rows

```
##                                          book              author
## 1                            Le Père Goriot      Honoré de Balzac
## 2   The Year of the Death of Ricardo Reis         José Saramago
## 3                               Aithiopika            Heliodorus
## 4                                   Shroud         John Banville
## 5             The Case of Comrade Tulayev          Victor Serge
## 6                   The 120 Days of Sodom       Marquis de Sade
## 7                            The Godfather            Mario Puzo
## 8                       The Drowned World          J.G. Ballard
## 9                          Go Down, Moses       William Faulkner
## 10                              Drop City  T. Coraghessan Boyle
```

Still not done

Need the nationalities of all the authors!

Pseduo code

Want a function to:

1. Read the wiki webpage using the author name.
2. Read the info card.
3. Get the nationality from the infocard.

If no nationality or infocard, want a function to:

1. Find which html paragraphs have text in them.
2. Guess the nationality from the text.
3. A function that bring the above all together.

More wrapping of code chunks

Want a function to read the wiki webpage using the author name

```
Read_wiki_page <- function(author_name){
  author_name_no_space = str_replace_all(author_name, "\\s+", "_")
  wiki_url <- paste("https://en.wikipedia.org/wiki/",
    author_name_no_space, sep = "")
  wiki_data <- read_html(wiki_url)
  return(wiki_data)
}
```

More wrapping of code chunks

Want a function to read the info card

```
Get_wiki_infocard <- function(wiki_data){
  infocard <- wiki_data |>
    rvest::html_nodes(".infobox.vcard") |>
    rvest::html_table(header = FALSE) |>
    purrr::pluck(1)
  return(infocard)
}
```

More wrapping of code chunks

Want a function to get the nationality from the infocard

```
Get_nationality_from_infocard <- function(infocard){
  nationality <- infocard %>%
    dplyr::rename(Category = X1, Response = X2) %>%
    dplyr::filter(Category == "Nationality") %>%
    dplyr::select(Response) %>%
    as.character()
  return(nationality)
}
```

More wrapping of code chunks

Need a function to find which html paragraphs have text in them

```
Get_first_text <- function(wiki_data){
  paragraph_data <- wiki_data %>%
    rvest::html_nodes("p")
  i = 1
  no_text = TRUE
  while(no_text){
    text_data <- paragraph_data %>%
      purrr::pluck(i) %>%
      rvest::html_text()
    check_text = gsub("\\s+", "", text_data)
    if(check_text == ""){
      i = i + 1
    }else{
      no_text = FALSE
    }
  }
```

More wrapping of code chunks

Need another function to get the nationality from the text

```
Guess_nationality_from_text <- function(text_data, possible_nationalities){

  num_matches <- str_count(text_data, possible_nationalities)
  prob_matches <- num_matches/sum(num_matches)

  i = which(prob_matches > 0)
  if(length(i) == 1){
    prob_nationality = possible_nationalities[i]
  }else if(length(i) > 0){
    warning(paste(c("More than one match for the nationality:",
                    possible_nationalities[i], "\n"), collapse = " "))
    match_locations = str_locate(text_data, possible_nationalities[i]) #gives lo
    j = i[which.min(match_locations[,1])]
    prob_nationality = possible_nationalities[j]
  }else{
    return(NA)
  }
```

More wrapping of code chunks

One function that brings that all together

```
Query_nationality_from_wiki <- function(author_name, possible_nationalities){

  wiki_data <- Read_wiki_page(author_name)

  infocard <- Get_wiki_infocard(wiki_data)

 if(is.null(infocard)){

    # nationality <- "Missing infocard"
   first_paragraph <- Get_first_text(wiki_data)
    nationality <- Guess_nationality_from_text(first_paragraph,
      possible_nationalities)

   return(nationality)

 }

 if(any(infocard[,1] == "Nationality")){
```

Examples

```
Query_nationality_from_wiki("Tim Winton", c("English", "British", "Australian"))

## [1] "Australian"

Query_nationality_from_wiki("Jane Austen" , c("English", "British", "Australian"

## [1] "English"

Query_nationality_from_wiki("Zadie Smith", c("English", "British", "Australian")

## [1] "English"
```

Still not done

We need a list of nationalities to search for in our text

What nationalities to search for?

```
# Get table of nationalities
url <- "http://www.vocabulary.cl/Basic/Nationalities.htm"
xpath <- "/html/body/div[1]/article/table[2]"
nationalities_df <- url %>%
  read_html() %>%
  html_nodes(xpath = xpath) %>%
  html_table() %>%
  as.data.frame()

possible_nationalities = nationalities_df[,2]
head(possible_nationalities)

## [1] "Afghan"                "Albanian"             "Algerian"
## [4] "ArgentineArgentinian" "Australian"           "Austrian"
```

Manual fixing

```
fix_entry = "ArgentineArgentinian"
i0 = which(nationalities_df == fix_entry, arr.ind = TRUE)
new_row = nationalities_df[i0[1], ]
nationalities_df[i0] = "Argentine"
new_row[,2] = "Argentinian"
nationalities_df = rbind(nationalities_df, new_row)

fix_footnote1 = "Colombia *"
i1 = which(nationalities_df == fix_footnote1, arr.ind = TRUE)
nationalities_df[i1] = strsplit(fix_footnote1, split = ' ')[[1]][1]

fix_footnote2 = "American **"
i2 = which(nationalities_df == fix_footnote2, arr.ind = TRUE)
nationalities_df[i2] = strsplit(fix_footnote2, split = ' ')[[1]][1]

possible_nationalities = nationalities_df[,2]

saveRDS(possible_nationalities, "data/possible_nationalities.rds")
```

Get Nationalities

```
nationality_from_author = sapply(book_data$author[1:20],
                                 function(author_name){
  nataionality = tryCatch( # Just in case!
     Query_nationality_from_wiki(author_name,
                                 possible_nationalities),
    error = function(e) NA)
  }) %>% unlist()

nationality_from_author
```

```
##                    Kazuo Ishiguro                         Ian McEwan
##                        "Japanese"                          "British"
##                       Zadie Smith                       J.M. Coetzee
##                         "English" "South AfricanAustralian (since 2006)"
##                     Peter Manson                      John Banville
##                        "Scottish"                          "Irish"
##                  Margaret Drabble                       Philip Roth
##                         "English"                          "American"
```

How diversely do we read

Run it!

```
nationality_from_author = sapply(book_data$author |> unique(),
                                 function(author_name){
  print(author_name)

  nationality = tryCatch( # Just in case!
     Query_nationality_from_wiki(author_name,
                                 possible_nationalities),
    error = function(e) NA)

})

author_nationality_df <- data.frame(
  author = book_data$author |> unique(),
  nationality = nationality_from_author)

book_data_with_nationality <- book_data |>
  dplyr::left_join(author_nationality_df)

head(book_data_with_nationality)
```

Result

```
## # A tibble: 6 × 2
##   nationality count
##   <chr>       <int>
## 1 <NA>          106
## 2 American       93
## 3 English        87
## 4 French         35
## 5 British        31
## 6 Irish          20
```

Let's take a look

```
library(plotly)
pie_plot <- table_nationalities %>%
  plot_ly(labels = ~nationality, values = ~count) %>%
  add_pie(hole = 0.6) %>%
  layout(title = "Nationalities",  showlegend = F,
         xaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE
         yaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE
```

Plotting result

```
pie_plot
```

A Few thoughts

- Many ways to approach this problem
- Approach here was to use the standard rvest toolbox
- Not perfect - much needed cleaning of nationality strings
- A bit of quessing of nationalities

What else could we have done

- Can scrape more data from goodreads website
- Goodreads has an API
- Check out the repository by famguy/rgoodreads to get started
- Using this API makes querying things like year or gender more straightforward
- But goodreads has no nationality, so this solution still is useful!

What else for webscraping

- There are easier ways to answer this same question
- Namely, RSelenium for pages with javascript
- Learning the hard way can be good sometimes though!

We should stop before we start scraping and think about whether we should.

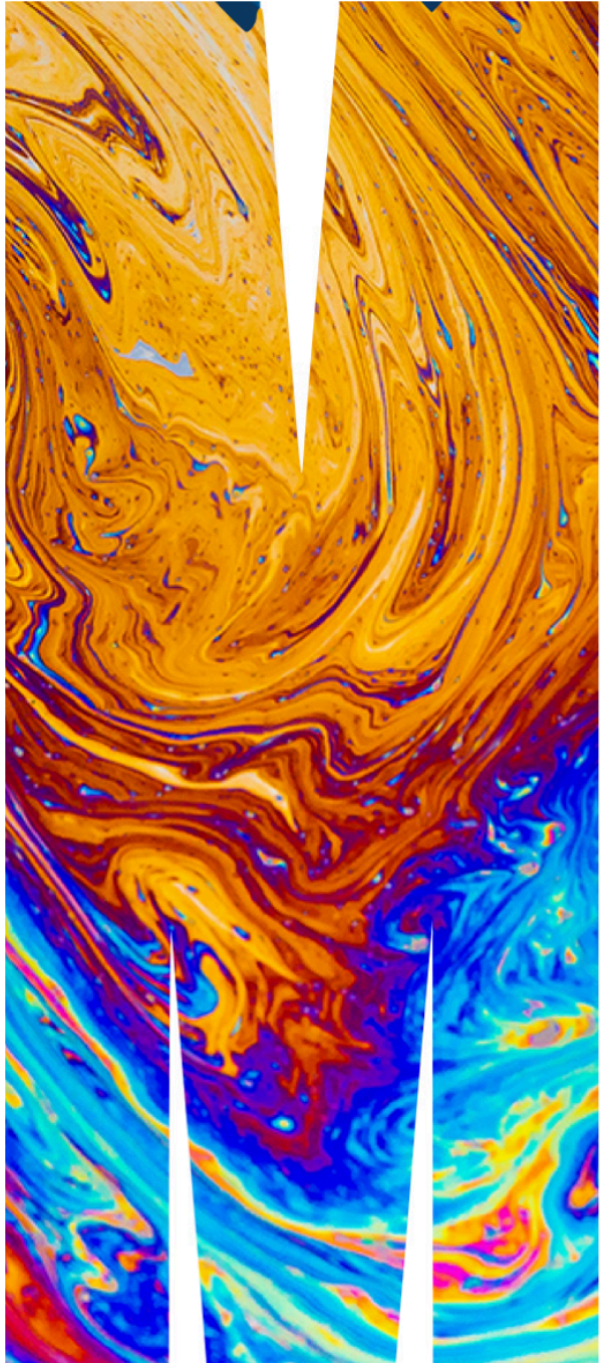
- Check the terms and conditions and terms of use
- Look for a data licence
- Consider ethics. Am I violating data privacy?
- Be considerate of the volume of queries and query rate limit
- Can look at the robots.txt file for the website

Summary

- We've learnt the basics of web scraping
- Know how to scrape from a statics website in R
- Work towards an automating our web scraping
- And we found out there are many challenges!

Slides developed by Dr Kate Saunders

MONASH University



Lecturer: *Kate Saunders*

Department of Econometrics and Business Statistics

ETC5512.Clayton-x@monash.edu

Week 12