Assignment 2: Solution

Karsten Donnay

15.06.20201

Contents

```
library(knitr)

### Global options
options(max.print="75")
opts_chunk$set(echo=FALSE,
                 cache=FALSE,
               prompt=FALSE,
               tidy=TRUE,
               comment=NA,
               message=FALSE,
               warning=FALSE)
opts_knit$set(width=75)
rm(list = ls())
```

Preparation

Install the 'stringr', 'xml2', and the 'jsonlite' package and load them.

```
# to_install <- c('stringr','xml2','jsonlite') install.packages(to_install)
library(xml2)
library(jsonlite)
library(stringr)
```

Assignment 1: Data extraction

Data type

In the Github repository (and on OLAT) under "assignment2/data/", you will find two files: file_a and file_b. Use a text editor of your choice, inspect each file, and determine what type of data each file includes.

- file_a: XML-file. Use xml2 package to load it.

- file_b: JSON-file. Use jsonlite package to to load it.

File A

Use an appropriate package to load the **file_a** into R. Try the functions associated with the package you used to load the data to get a feeling for the dataset and extract the following information with it.

- Extract a list of the IDs of the books.
- What is the name of the author of the 4th book?

```r
rm(list = ls())
books_data <- xml2::read_xml("data/file_a")

# Get the book IDs
(ids <- books_data %>%
    xml_children %>%
    xml_attrs %>%
    unlist)
```

```
     id      id      id      id      id      id      id      id      id      id
"bk101" "bk102" "bk103" "bk104" "bk105" "bk106" "bk107" "bk108" "bk109" "bk110"
     id      id
"bk111" "bk112"
```

```r
# Get authors
authors_ind <- books_data %>%
    xml_children %>%
    xml_children %>%
    xml_name %>%
    grep(pattern = "author", x = .)
(authors <- (books_data %>%
    xml_children %>%
    xml_children %>%
    xml_text)[authors_ind])[4]
```

```
[1] "Corets, Eva"
```

File B

Use the appropriate package to load **file_b** into R.

- What is the email of Mr. Bea?

```r
rm(list = ls())
adresses <- jsonlite::fromJSON("data/file_b")

# Base R
adresses[adresses$last_name == "Bea", ]$email
```

```
[1] "nbea2@imageshack.us"
```

```r
# Or with dplyr
adresses %>%
    dplyr::filter(last_name == "Bea") %>%
    dplyr::select(email)
```

```
                email
1 nbea2@imageshack.us
```

Assignment 2: Extract dates with Regular Expressions

You have the URL to an article of the Guardian. Use regular expressions to extract the publication date of the linked article. Your procedure should be generally applicable to the articles of The Guardian and not only work for this one link. Use the package **stringr** and its function **str_extract** or optionally **str_replace** to solve this task. (Hint: Slashes and Backslashes must be escaped with two preceding backslashes; e.g. ‘ /’ or ’ ’. The package lubridate and its function as_date() are helpful when trying to transform text to dates.)

Multiple ways:

- Remove everything around the feature of interest until only feature is left (e.g. stringr::str_replace)
- Extract only feature of interest (e.g. stringr::str_extract)

(a) Removal

```
url <- "https://www.theguardian.com/society/2018/oct/11/new-law-employers-reveal-race-pay-gap-figures"
# Structure: https://www.theguardian.com/category/year/month/day/headlinetext
```

1. Remove domain

```
stringr::str_extract(url, "https:\\/\\/www.theguardian.com\\/")  # Matches https://www.theguardian.com/
```

```
[1] "https://www.theguardian.com/"
```

```
(new <- stringr::str_replace(string = url, pattern = "https:\\/\\/www.theguardian.com\\/",
    replacement = ""))
```

```
[1] "society/2018/oct/11/new-law-employers-reveal-race-pay-gap-figures"
```

2. Remove category + /

```
stringr::str_extract(new, "[:alpha:]*\\/")  # Matches any letter before a slash and the slash itself
```

```
[1] "society/"
```

```
(new <- stringr::str_replace(string = new, pattern = "[:alpha:]*\\/", replacement = ""))
```

```
[1] "2018/oct/11/new-law-employers-reveal-race-pay-gap-figures"
```

3. Remove / + ending

```
stringr::str_extract(new, "\\/[[:lower:]+-]+$")  # Matches a slash and text with hyphens (at least one
```

```
[1] "/new-law-employers-reveal-race-pay-gap-figures"
```

```
(new <- stringr::str_replace(string = new, pattern = "\\/[[:lower:]+-]+$", replacement = ""))
```

```
[1] "2018/oct/11"
```

```
# 4. Convert to date with lubridate
(date <- lubridate::as_date(new))
```

```
[1] "2018-10-11"
```

(b) Extraction

- Idea: extract everything between the first occurrences of a ‘/NUMBER’ (year) to ‘NUMBER/’ (day)
- Maybe important: Days also end with two numbers even if they are a single number (e.g. 2020/apr/07)

- Also: Months are all represented by three letters

```
url <- "https://www.theguardian.com/society/2018/oct/11/new-law-employers-reveal-race-pay-gap-figures"
# Structure: https://www.theguardian.com/category/year/month/day/headlinetext
# 1. Extract pattern: 4*digits/3*letters/2*digits
(new <- stringr::str_extract(string = url, pattern = "[:digit:]{4}\\/[:lower:]{3}\\/[:digit:]{2}"))
```

```
[1] "2018/oct/11"
```

```
# 2. Convert to date with lubridate
(date <- lubridate::as_date(new))
```

```
[1] "2018-10-11"
```

Assignment 3: Translation

Take the content of file_b used in the first assignment. Transform it into an XML-File format. You can use https://www.xmlvalidation.com to validate your attempts.

Solution: The file should look something just like this. You can choose any name instead of 'adresses' or 'contact'.

```
<adresses>
  <contact id="1">
    <first_name>Jeanette</first_name>
    <last_name>Penddreth</last_name>
    <gender>Female</gender>
    <ip_adress>23.58.193.2</ip_adress>
  </contact>
  <contact id="2">
    ...
  </contact>
  ...
</adresses>
```