Loading from JSON, HTML, XML

By Tony Fraser

2023-10-19

Assignment

Use R to load data about books from HTML, JSON and XML. Then compare.

Explanation: each of these books has more than one author, and therefore this data is not inherently flat. This could be transformed and loaded into a tidy data frame, but there are other data types better suited for this structure. In R, this data is better represented by a list of lists, in python a dictionary, in scala a Map[String,Array] type, etc.

We are going to do this exercise in list of list format, but before we do that let's first look at what this data looks like if explicitly declared.

Load from list of lists

```
books_list <- list(
  list( title = "The Wildest Dream: The Biography of George Mallory",
        authors = c("Peter Gillman", "Leni Gillman"),
        description = "This biography delves ... disappeared during .. 1924 ... ",
        year_published = 2000
  ),
  list( title = "K2: Life and Death on the World's Most Dangerous Mountain",
        authors = c("Ed Viesturs", "David Roberts"),
        description = "This book ... allure of K2, the world's second-highest mountain..."
        year_published = 2009
  )
  ## there are four total records in actual samples, this is just for a quick visual.
)
## Access and test
# > books[[1]]$authors[[2]]
#[1] "Leni Gillman"
```

Load from JSON

Most all languages have a loader that goes straight from JSON into a dictionary type data set.
In R, it's called `jsonlite`.

Four mountaineering books as JSON

```
json_path = paste(base_path, "week7/mountains.json", sep = "")
library(jsonlite)

# not this!!  This goes straight into  a dataframe. We don't want this additional
# "simplify" processing, we want to keep it in list[list] form.
# books <- fromJSON(readLines(json_path, warn = FALSE))

# use this one!
books_json <- fromJSON(readLines(json_path, warn = FALSE), simplifyDataFrame = FALSE)

## Access and test
#> books_json[[1]]$authors[[2]]
#[1] "Leni Gillman"
```

Load from XML

Four mountaineering books as XML

There is an R xml parser binary called xml2. xml2 is an important package, but it is important mostly because it is used as the engine that powers the rvest package. rvest is a critically important package that scrapes websites, extracts data from HTML DOM's and XPaths, reads through CSS's, etc. For the purpose of this lab, it also extracts from both HTML and XML.

```
library(rvest)
xml_parsed <- read_html(paste(base_path, "week7/mountains.xml", sep = "") )

titles <- xml_parsed %>% html_nodes(xpath = ".//title") %>% html_text()

# Extract authors
authors_nodes <- xml_parsed %>% html_nodes(xpath = ".//authors")
authors <- lapply(authors_nodes, function(node) {
  node %>% html_nodes(xpath = ".//author") %>% html_text()
})

descriptions <- xml_parsed %>% html_nodes(xpath = ".//description") %>% html_text()
years <- xml_parsed %>% html_nodes(xpath = ".//year_published") %>% html_text() %>% as.num

books_xml <- lapply(1:length(titles), function(i) {
  list(
    title = titles[i],
    authors = authors[[i]],
    description = descriptions[i],
    year_published = as.integer(years[i])
  )
})
## Access and test
# > books_xml[[1]]$authors[[2]]
# [1] "Leni Gillman"
```

Load from HTML

R's Rvest is paraser that that can extract content out of HTML files. If you want to find all the links on a page, or extract all the books in a DOM, or an XPath, rvest is what you are looking for.

Four mountaineering books as HTML

```
library(rvest)
html_parsed <- read_html(paste(base_path, "week7/mountains.html", sep = "") )
titles <- html_parsed %>%
  html_nodes(".book h2") %>% html_text()
authors_texts <- html_parsed %>%
  html_nodes(xpath = "//div[@class='book']/p[1]") %>% html_text()
authors <- strsplit(gsub("Authors: ", "", authors_texts), ", ")

descriptions <- html_parsed %>%
  html_nodes(xpath = "//div[@class='book']/p[2]") %>% html_text()
years_texts <- html_parsed %>%
  html_nodes(xpath = "//div[@class='book']/p[3]") %>% html_text()
years <- as.numeric(gsub("Year Published: ", "", years_texts))

books_html <- lapply(1:length(titles), function(i) {
  list(
    title = titles[i],
    authors = authors[[i]],
    description = descriptions[i],
    year_published = as.integer(years[i])
  )
})

## Access and test
# > books_html[[1]]$authors[[2]]
#[1] "Leni Gillman"
```

Are they identical?

They certainly weren't when I started this project, but they are now.

```
identical_books_json_html <- identical(books_json, books_html)
identical_books_html_xml <- identical(books_html, books_xml)
identical_books_json_xml <- identical(books_json, books_xml)

all_identical <- identical_books_json_html && identical_books_html_xml && identical_books_

if (all_identical) {
    print("Yes, they are identical.")
} else {
    print("No, they are not identical.")
}
```

```
[1] "Yes, they are identical."
```