# Webscraping with RSelenium

Automate your browser actions

Etienne Bacher

LISER

2022-09-20

## Introduction

Do you really need scraping?

Before scraping: is there an API?

- if yes, is there a package?
    - if yes, use the package
    - if no, build the API queries yourself with `{httr}`
- if no, scrape (politely)

## Introduction

Scraping can be divided in two steps:

1. getting the HTML that contains the information

## Introduction

Scraping can be divided in two steps:

1. getting the HTML that contains the information
2. cleaning the HTML to extract the information we want

## Introduction

Scraping can be divided in two steps:

1. getting the HTML that contains the information
2. cleaning the HTML to extract the information we want

## Introduction

Scraping can be divided in two steps:

1. getting the HTML that contains the information
2. cleaning the HTML to extract the information we want

These 2 steps don't necessarily require the same tools, and *shouldn't be made at the same time*.

## Introduction

Here, we will focus on the first step: **how to obtain the HTML code you need on dynamic pages?**

## Static and dynamic pages

## Static and dynamic pages

The web works with 3 languages:

- HTML: content and structure of the page

## Static and dynamic pages

The web works with 3 languages:

- HTML: content and structure of the page
- CSS: style of the page

## Static and dynamic pages

The web works with 3 languages:

- HTML: content and structure of the page
- CSS: style of the page
- JavaScript: interactions with the page

## Static and dynamic pages

The web works with 3 languages:

- HTML: content and structure of the page
- CSS: style of the page
- **JavaScript**: interactions with the page

## Static vs dynamic

**Static webpage**:

- all the information is loaded with the page;
- changing a parameter modifies the URL

Examples: Wikipedia, IMDB.

## Static vs dynamic

**Static webpage**:

- all the information is loaded with the page;
- changing a parameter modifies the URL

Examples: Wikipedia, IMDB.

**Dynamic webpage**: the website uses JavaScript to fetch data from their server and *dynamically* update the page.

Example: see later.

Why is it harder to do webscraping with dynamic pages?

Webscraping a static website can be quite simple:

- you get a list of URLs;
- download the HTML for each of them;
- read and clean the HTML

and that's it.

Webscraping a static website can be quite simple:

- you get a list of URLs;
- download the HTML for each of them;
- read and clean the HTML

and that's it.

This is easy because you can differentiate two pages with different content just by looking at their URL.

Example:

But in dynamic pages, there's no obvious way to see that the inputs are different:

So it seems that the only way to get the data is to go manually through all pages to get the HTML.

So it seems that the only way to get the data is to go manually through all pages to get the HTML.

*350h and 3 RAs later...*

## (R)Selenium

## Idea

Idea: control the browser from the command line.

## Idea

Idea: control the browser from the command line.

*"I wish I could click on this button to open a modal"*

```
remote_driver$
  findElement(using = "css", value = ".my-button")$
  clickElement()
```

## Idea

Idea: control the browser from the command line.

*“I wish I could click on this button to open a modal”*

```
remote_driver$
  findElement(using = "css", value = ".my-button")$
  clickElement()
```

*“I wish I could fill these inputs to automatically connect”*

```
remote_driver$
  findElement(using = "id", value = "password")$
  sendKeysToElement(list("my_super_secret_password"))
```

Almost everything you can do “by hand” in a browser, you can reproduce with Selenium:

Almost everything you can do “by hand” in a browser, you can reproduce with Selenium:

| | |
|---|---|
| ▶ open a browser | ▶ `open()` / `navigate()` |
| ▶ click on something | ▶ `clickElement()` |
| ▶ enter values | ▶ `sendKeysToElement()` |
| ▶ go to previous/next page | ▶ `goBack()` / `goForward()` |
| ▶ refresh the page | ▶ `refresh()` |
| ▶ get all the HTML that is currently displayed | ▶ `getPageSource()` |

## Get started

## Get started

In the beginning there was ~~light~~ `rsDriver()`:

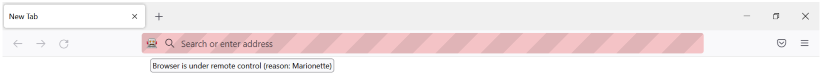
```
# if not already installed
# install.packages("RSelenium")
library(RSelenium)

driver <- rsDriver(browser = "firefox") # can also be chron
remote_driver <- driver[["client"]]
```

## Get started

In the beginning there was ~~light~~ `rsDriver()`:

```
# if not already installed
# install.packages("RSelenium")
library(RSelenium)

driver <- rsDriver(browser = "firefox") # can also be chron
remote_driver <- driver[["client"]]
```

This will print a bunch of messages and open a “marionette browser”.



## Get started

From now on, everything we do is calling `<function>()` starting with `remote_driver$`[1].

[1] Or whatever you called it in the previous step.

## Exercise 1

## Exercise 1

**Objective:** get the list of core contributors to R located here.

## Exercise 1

**Objective:** get the list of core contributors to R located here.

How would you do it by hand?

- open the browser;
- go to https://r-project.org;
- in the left sidebar, click on the link “Contributors”;
- and voilà!

## Exercise 1

**Objective:** get the list of core contributors to R located here.

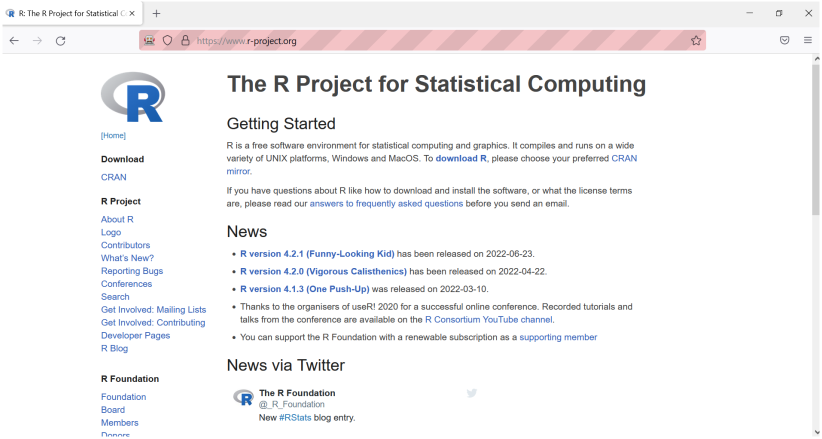
How would you do it by hand?

- open the browser;
- go to https://r-project.org;
- in the left sidebar, click on the link “Contributors”;
- and voilà!

How can we do these steps programmatically?

## Open the browser and navigate

```r
remote_driver$navigate("https://r-project.org")
```



# The R Project for Statistical Computing

[Home]

**Download**

CRAN

**R Project**

About R
Logo
Contributors
What's New?
Reporting Bugs
Conferences
Search
Get Involved: Mailing Lists
Get Involved: Contributing
Developer Pages
R Blog

**R Foundation**

Foundation
Board
Members
Donors

## Getting Started

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To **download R**, please choose your preferred CRAN mirror.

If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

## News

- **R version 4.2.1 (Funny-Looking Kid)** has been released on 2022-06-23.
- **R version 4.2.0 (Vigorous Calisthenics)** has been released on 2022-04-22.
- **R version 4.1.3 (One Push-Up)** was released on 2022-03-10.
- Thanks to the organisers of useR! 2020 for a successful online conference. Recorded tutorials and talks from the conference are available on the R Consortium YouTube channel.
- You can support the R Foundation with a renewable subscription as a supporting member

## News via Twitter

**The R Foundation**
@_R_Foundation
New #RStats blog entry.

## Click on “Contributors”

This requires two things:

1. find the element
2. click on it

**How to find an element?**

- Humans -> eyes
- Computers -> HTML/CSS

To find the element, we need to open the console to see the structure of the page.

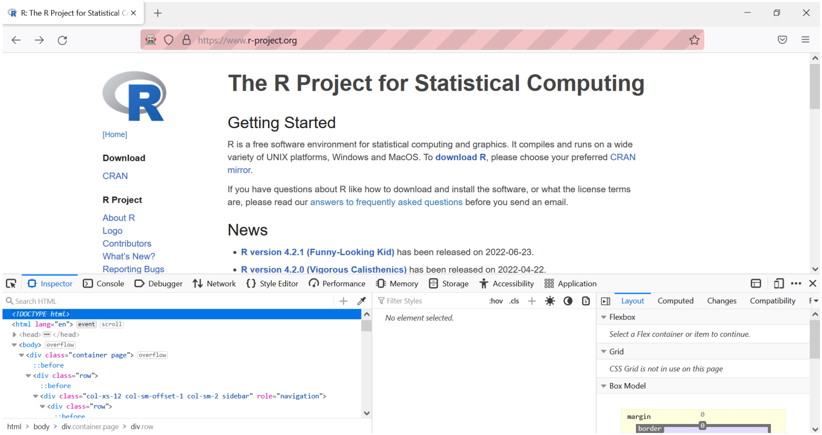
Several ways to do it:

- right-click -> “Inspect”
- `Ctrl` + `Shift` + `C`

To find the element, we need to open the console to see the structure of the page.

Several ways to do it:

- right-click -> "Inspect"
- `Ctrl + Shift + C`



The R Project for Statistical Computing

Getting Started

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To download R, please choose your preferred CRAN mirror.

If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

News

- R version 4.2.1 (Funny-Looking Kid) has been released on 2022-06-23.
- R version 4.2.0 (Vigorous Calisthenics) has been released on 2022-04-22.

[Home]

Download

CRAN

R Project

About R
Logo
Contributors
What's New?
Reporting Bugs

```
<!DOCTYPE html>
<html lang="en"> event scroll
  <head> ··· </head>
  <body> overflow
    <div class="container page"> overflow
      ::before
      <div class="row">
        ::before
        <div class="col-xs-12 col-sm-offset-1 col-sm-2 sidebar" role="navigation">
          <div class="row">
            ::before
```

No element selected.

Flexbox

Select a Flex container or item to continue.

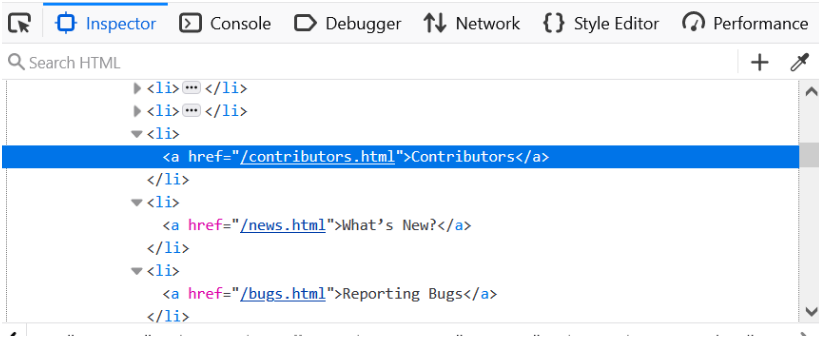
Grid

CSS Grid is not in use on this page

Box Model

margin 0
border 0

Then, hover the element we're interested in: the link "Contributors".



```
<li>⋯</li>
<li>⋯</li>
<li>
  <a href="/contributors.html">Contributors</a>
</li>
<li>
  <a href="/news.html">What's New?</a>
</li>
<li>
  <a href="/bugs.html">Reporting Bugs</a>
</li>
```

How can we find this with `RSelenium`?

```
?RSelenium::remoteDriver
```

-> `findElement`

- class name
- id
- name
- tag name
- css selector
- link text
- partial link text
- xpath

All of these work:

```
remote_driver$
  findElement("link text", "Contributors")$
  clickElement()

remote_driver$
  findElement("partial link text", "Contributors")$
  clickElement()

remote_driver$
  findElement("xpath", "/html/body/div/div[1]/div[1]/div/di
  clickElement()

remote_driver$
  findElement("css selector", "div.col-xs-6:nth-child(1) >
  clickElement()
```

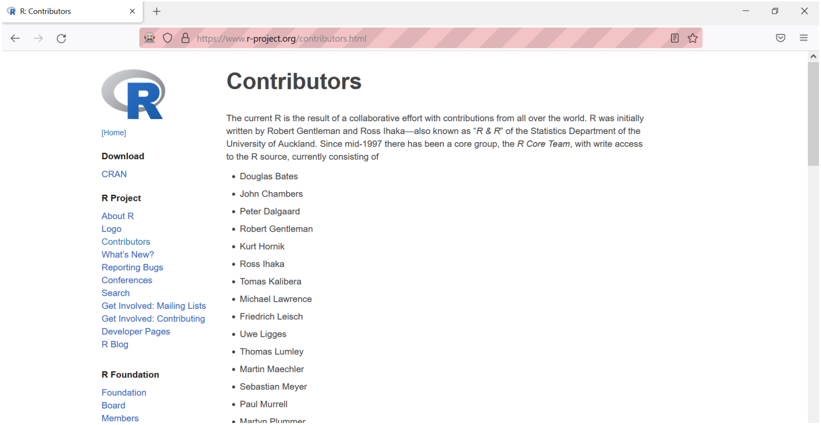
All of these work:

```
remote_driver$
  findElement("link text", "Contributors")$
  clickElement()

remote_driver$
  findElement("partial link text", "Contributors")$
  clickElement()

remote_driver$
  findElement("xpath", "/html/body/div/div[1]/div[1]/div/d
  clickElement()

remote_driver$
  findElement("css selector", "div.col-xs-6:nth-child(1) >
  clickElement()
```

Tip

You can check that you found the right element by highlighting

## We are now on the right page!



# Contributors

The current R is the result of a collaborative effort with contributions from all over the world. R was initially written by Robert Gentleman and Ross Ihaka—also known as "*R & R*" of the Statistics Department of the University of Auckland. Since mid-1997 there has been a core group, the *R Core Team*, with write access to the R source, currently consisting of

- Douglas Bates
- John Chambers
- Peter Dalgaard
- Robert Gentleman
- Kurt Hornik
- Ross Ihaka
- Tomas Kalibera
- Michael Lawrence
- Friedrich Leisch
- Uwe Ligges
- Thomas Lumley
- Martin Maechler
- Sebastian Meyer
- Paul Murrell
- Martyn Plummer

[Home]

**Download**

CRAN

**R Project**

About R
Logo
Contributors
What's New?
Reporting Bugs
Conferences
Search
Get Involved: Mailing Lists
Get Involved: Contributing
Developer Pages
R Blog

**R Foundation**

Foundation
Board
Members

Last step: obtain the HTML of the page.

```
remote_driver$getPageSource()
```

Last step: obtain the HTML of the page.

```r
remote_driver$getPageSource()
```

To read it with `rvest`:

```r
x <- remote_driver$getPageSource()[[1]]
rvest::read_html(x)
```

Do we read the HTML and extract the information in the same script?

Do we read the HTML and extract the information in the same script?

**No!**

Instead, we save the HTML in an external file, and we will be able to access it in another script (and offline) to manipulate it as we want.

```r
write(x, file = "contributors.html")
# Later and in another script
rvest::read_html("contributors.html")
```

Click here to see the results.

## Exercise 2: a harder & real-life example

The previous example was not a *dynamic* page: we could have used the link to the page and apply webscraping methods for static webpages.

```r
rvest::read_html("https://www.r-project.org/contributors.ht
```

The previous example was not a *dynamic* page: we could have used the link to the page and apply webscraping methods for static webpages.

```r
rvest::read_html("https://www.r-project.org/contributors.ht
```

Let's now dive into a more complex example, where RSelenium is the only way to obtain the data.

## Before using RSelenium

*Using RSelenium is slower than using "classic" scraping methods*, so it's important to check all possibilities before using it.

Use Selenium if:

- the HTML you want is not directly accessible, i.e needs some interactions (clicking on a button, connect to a website...)
- the URL doesn't change with the inputs
- you can't access the data directly in the "network" tab of the console

## Before using RSelenium

*Using RSelenium is slower than using “classic” scraping methods*, so it’s important to check all possibilities before using it.

Use Selenium if:

- the HTML you want is not directly accessible, i.e needs some interactions (clicking on a button, connect to a website...)
- the URL doesn’t change with the inputs
- you can’t access the data directly in the “network” tab of the console

Interesting read: the Ethical Scraper

## Example: Sao Paulo immigration museum

ASK MARTIN FIRST

## Appendix

## Appendix

For reference, here's the code to extract the list of contributors:

```
library(rvest)

html <- read_html("contributors.html")

bullet_points <- html %>%
  html_elements(css = "div.col-xs-12 > ul > li") %>%
  html_text()

blockquote <- html %>%
  html_elements(css = "div.col-xs-12.col-sm-7 > blockquote'
  html_text() %>%
  strsplit(., split = ", ")

blockquote <- blockquote[[1]] %>%
  gsub("\\r|\\n|\\.|and", "", .)

others <- html %>%
```

# Appendix

```
 [1] "Douglas Bates"          "John Chambers"          "Pe
 [4] "Robert Gentleman"       "Kurt Hornik"            "Ro
 [7] "Tomas Kalibera"         "Michael Lawrence"       "Fr
[10] "Uwe Ligges"             "Thomas Lumley"          "Ma
[13] "Sebastian Meyer"        "Paul Murrell"           "Ma
[16] "Brian Ripley"           "Deepayan Sarkar"        "Du
[19] "Luke Tierney"           "Simon Urbanek"          "Va
[22] "Suharto Anggono"        "Thomas Baier"           "Ga
[25] "Henrik Bengtsson"       "Roger Biv"              "Be
[28] "David Brahm"            "Göran Broström"         "Pa
[31] "Vince Carey"            "Saikat DebRoy"          "Ma
[34] "Brian D'Urso"           "Lyndon Drake"           "Di
[37] "Claus Ekstrom"          "Sebastian Fischmeister" "Jo
[40] "Paul Gilbert"           "Yu Gong"                "Ga
[43] "Frank E Harrell Jr"     "Peter M Haverty"        "To
[46] "Robert King"            "Kjetil Kjernsmo"        "Ro
[49] "Philippe Lambert"       "Jan de Leeuw"           "Ji
[52] "Patrick Lindsey"        "Catherine Loader"       "Go
```