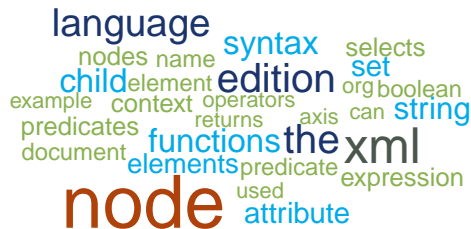


Introduction to Web Scraping with R

Using SelectorGadget



Simon Munzert | IPSDS

SelectorGadget

SelectorGadget

Scraping webpages

- the most cumbersome part of web scraping (data tidying aside) is the construction of XPath expressions that match the components of a page you want to extract
- it will take a couple of scraping projects until you'll truly have mastered XPath

SelectorGadget

Scraping webpages

- the most cumbersome part of web scraping (data tidying aside) is the construction of XPath expressions that match the components of a page you want to extract
- it will take a couple of scraping projects until you'll truly have mastered XPath

A much-appreciated helper

- **SelectorGadget** is a JavaScript browser plugin that constructs XPath statements (or CSS selectors) via a point-and-click approach
- it is available here: <http://selectorgadget.com/> (there s also a Chrome extension)
- the tool is magic and you will love it

What does SelectorGadget do?

What does SelectorGadget do?

On the surface

1. you activate the tool on any webpage you want to scrape
2. based on your selection of components, the tool learns about your desired components and generates an XPath expression (or CSS selector) for you

What does SelectorGadget do?

On the surface

1. you activate the tool on any webpage you want to scrape
2. based on your selection of components, the tool learns about your desired components and generates an XPath expression (or CSS selector) for you

Under the hood

- based on your selection(s), the tool looks for similar elements on the page
- the underlying algorithm, which draws on Google's diff-match-patch libraries, focuses on CSS characteristics, such as tag names and `<div>` and `` attributes

Installation and use

Installation

Go to <http://selectorgadget.com/> and drag the link provided at the end of the page ("Or drag this link to your bookmark bar") to your bookmark bar.

Installation and use

Installation

Go to <http://selectorgadget.com/> and drag the link provided at the end of the page ("Or drag this link to your bookmark bar") to your bookmark bar.

Use

1. open the webpage you want to scrape
2. click on the SelectorGadget bookmarklet
3. click on the elements you want to extract. Manually selected elements will turn **green**. Elements that match the selector will be highlighted **yellow**
4. de-select the elements you do not want to extract - they will turn **red**.
5. repeat steps 3–4 until the marked set of elements matches the set you want to extract
6. click on the "XPath" button in the console
7. copy the generated XPath expression and process it with **rvest**

Example

Example

Example

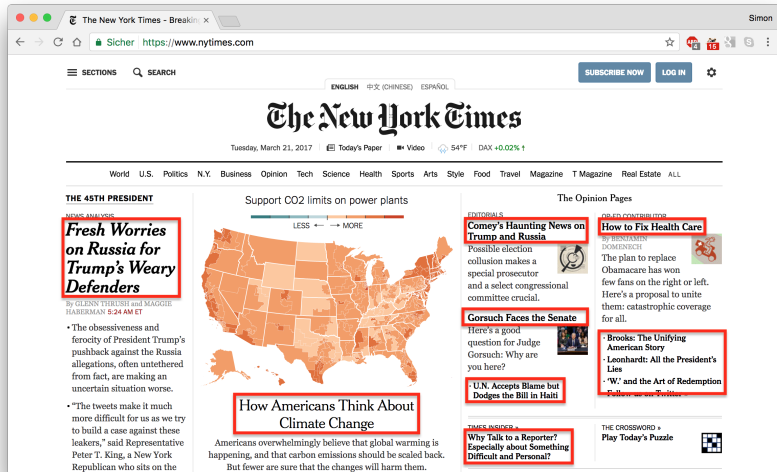
- source:
<https://nytimes.com>



Example

Example

- source:
<https://nytimes.com>
- goal: scrape all headlines



Example

Example

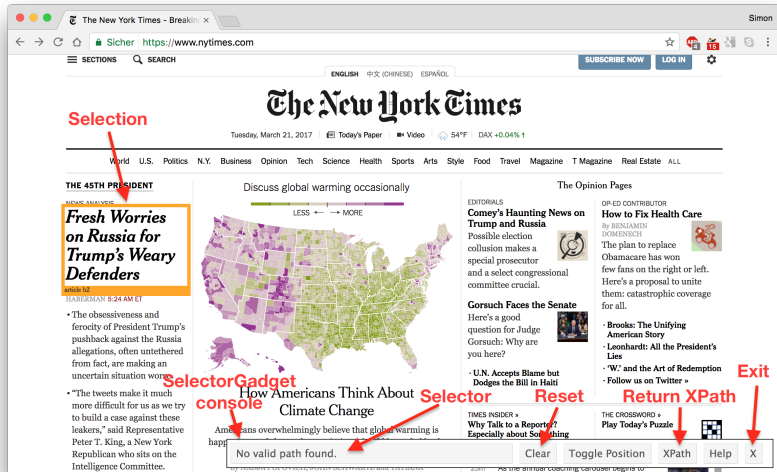
1. click on SelectorGadget bookmarklet (not shown)
2. click on first headline



Example

Example

1. click on SelectorGadget bookmarklet (not shown)
2. click on first headline



Example

Example

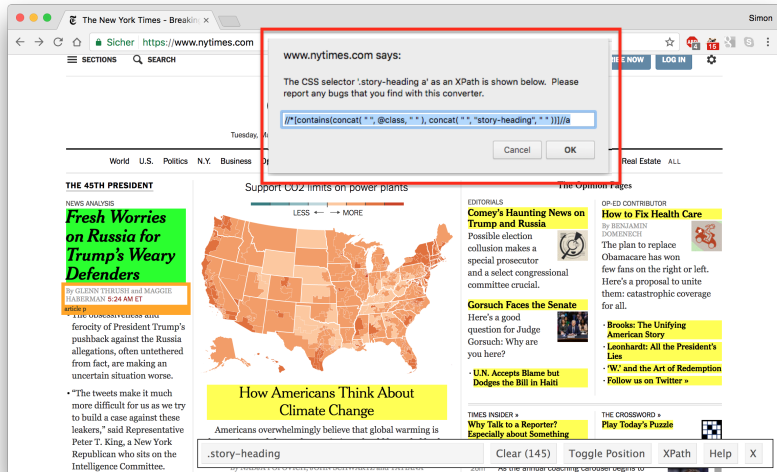
1. click on SelectorGadget bookmarklet (not shown)
2. click on first headline
3. all relevant headlines selected



Example

Example

1. click on SelectorGadget bookmarklet (not shown)
2. click on first headline
3. all relevant headlines selected
4. click on "XPath"
5. copy proposed expression



Example

Now, we switch to R

R code

```
1 library(rvest)
2 url_p <- read_html("https://www.nytimes.com")
3 headlines <- html_nodes(url_p, xpath = '//*[@contains(concat( " ", @class, " " ), concat( " ",
  "story-heading", " " ))]//a') # we use single quotation marks here to wrap around the
  expression!
4 headlines_raw <- html_text(headlines)
5 head(headlines_raw)

[1] "Fresh Worries on Russia for Trump's Weary Defenders"
[2] "F.B.I. Confirms Inquiry on Trump Team's Russia Ties"
[3] "Takeaways From the Hearing "
[4] "\n      Trump and the Russians: Links? No Links?"
[5] "G.O.P. Responds by Changing Subject"
[6] "U.S. Limits Devices on Foreign Airlines From 8 Countries"
```

end

Example

Let's clean the data

R code

```
6 headlines_clean <- headlines_raw %>% str_replace_all("\\n", "") %>% str_trim()
7 length(headlines_clean)
[1] "Fresh Worries on Russia for Trump's Weary Defenders"
[2] "F.B.I. Confirms Inquiry on Trump Team's Russia Ties"
[3] "Takeaways From the Hearing"
[4] "Trump and the Russians: Links? No Links?"
[5] "G.O.P. Responds by Changing Subject"
[6] "U.S. Limits Devices on Foreign Airlines From 8 Countries"
8 str_detect(headlines_clean, "Trump") %>% table()
FALSE  TRUE
  133    12
```

end

Summary

Summary

So why did I bother you with learning XPath at all?

Summary

So why did I bother you with learning XPath at all?

Caveats

- SelectorGadget is not perfect. Sometimes, the algorithm will fail in finding a useful XPath expression
- starting from a different element sometimes (but not always!) helps
- often the generated expressions are unnecessarily complex, unintuitive and therefore difficult to debug
- in my experience, SelectorGadget works 70-80% of the times when you want to scrape info from a static webpage
- you are also prepared for the remaining 20-30%!



Source: http://inspectorgadget.wikia.com/wiki/File:Inspector_Gadget_Thinking.png
(DANTHEMAN123)