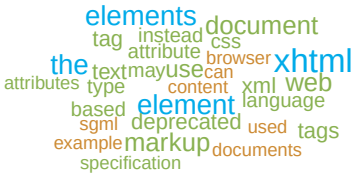
Introduction to Web Scraping with R

Inspecting the HTML Tree



Simon Munzert | IPSDS

Browsing vs. scraping

Browsing vs. scraping the web

Browsing vs. scraping the web

Using your browser to access webpages

1. you click on a link, enter a URL, run a Google query, etc.
2. browser/your machine sends request to server that hosts website
3. server returns resource (often an HTML document)
4. browser interprets HTML and renders it in a nice fashion

Browsing vs. scraping the web

Using your browser to access webpages

1. you click on a link, enter a URL, run a Google query, etc.
2. browser/your machine sends request to server that hosts website
3. server returns resource (often an HTML document)
4. browser interprets HTML and renders it in a nice fashion

Using R to access webpages

1. you manually specify a resource
2. R/your machine sends request to server that hosts website
3. server returns resource
4. R parses HTML, but does not render it in a nice fashion
5. it's up to you to tell R what content to extract

Interacting with your browser

On web browsers

- modern browsers are complex pieces of software that take care of multiple operations while you browse the web
- common operations: retrieve resources, render and display information, provide interface for user-webpage interaction
- although your goal is to automate web data retrieval, your browser is an important tool in web scraping workflow

Interacting with your browser

The use of browsers for web scraping

- give you an intuitive impression of the architecture of a webpage
- allow you to inspect the source code
- let you construct XPath/CSS selector expressions with plugins
- render dynamic web content (JavaScript interpreter)

Interacting with your browser

The use of browsers for web scraping

- give you an intuitive impression of the architecture of a webpage
- allow you to inspect the source code
- let you construct XPath/CSS selector expressions with plugins
- render dynamic web content (JavaScript interpreter)

A note on browser differences

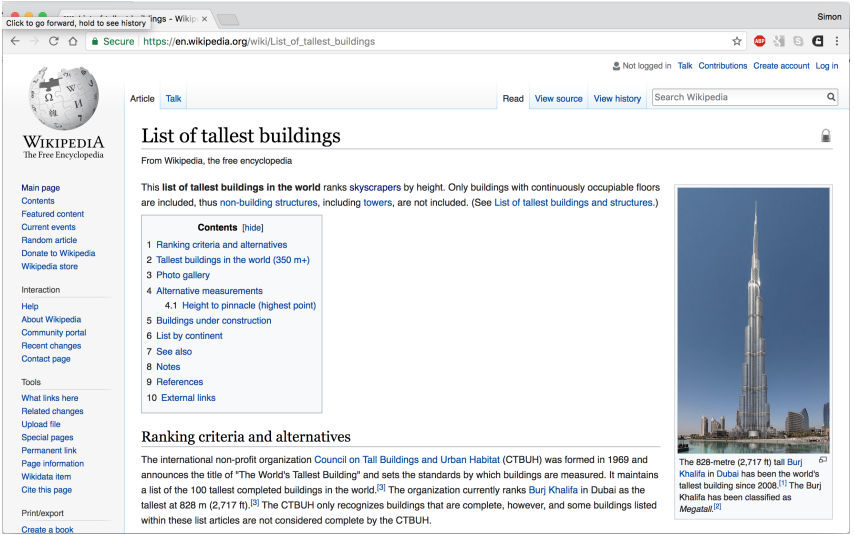
- inspecting the source code (as shown on the following slides) works more or less identically in **Chrome** and **Firefox**
- in **Safari**, go to → `Preferences`, then → `Advanced` and select "Show Develop menu in menu bar". This unlocks the "Show Page Source" option and the Web Developer Tools

Inspecting the HTML source code

Inspecting the HTML source code

Example

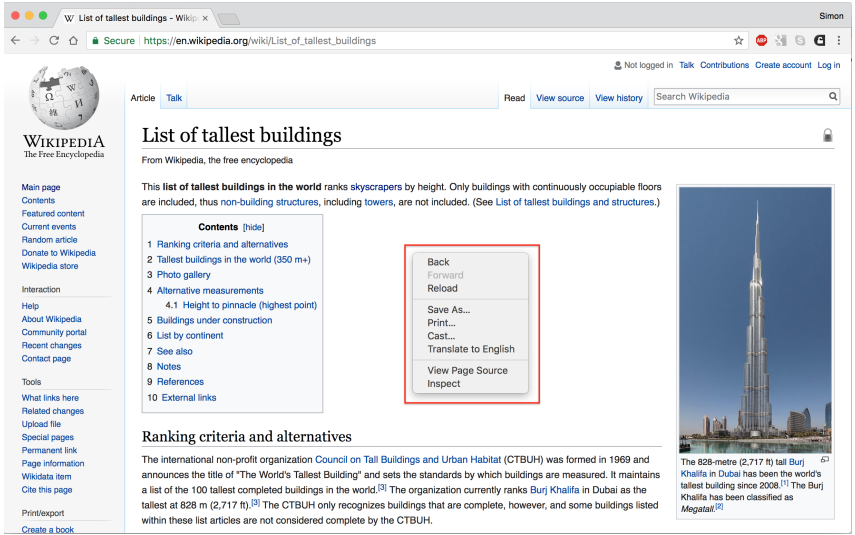
- browser: Google Chrome
- source: https://en.wikipedia.org/wiki/List_of_tallest_buildings



Inspecting the HTML source code

Example

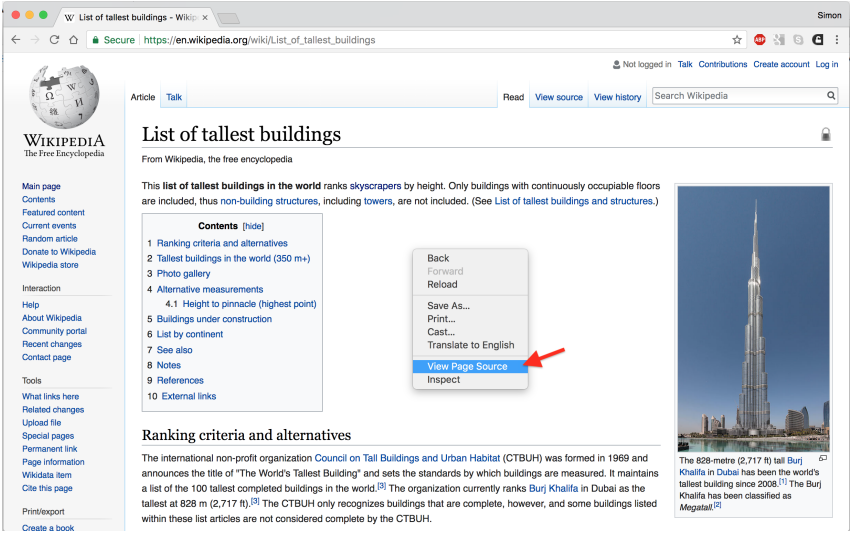
- browser: Google Chrome
- source: https://en.wikipedia.org/wiki/List_of_tallest_buildings
- right-click on page



Inspecting the HTML source code

Example

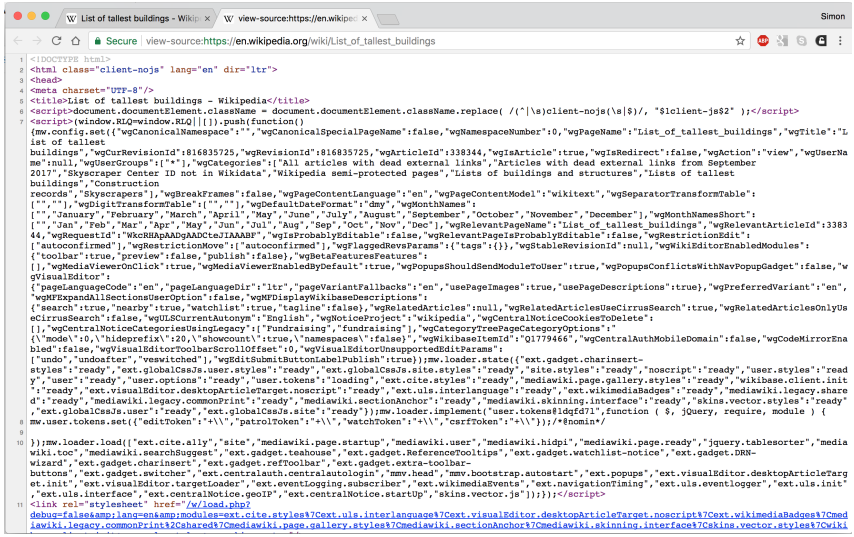
- browser: Google Chrome
- source: https://en.wikipedia.org/wiki/List_of_tallest_buildings
- right-click on page
- select "View Page Source"



Inspecting the HTML source code

Example

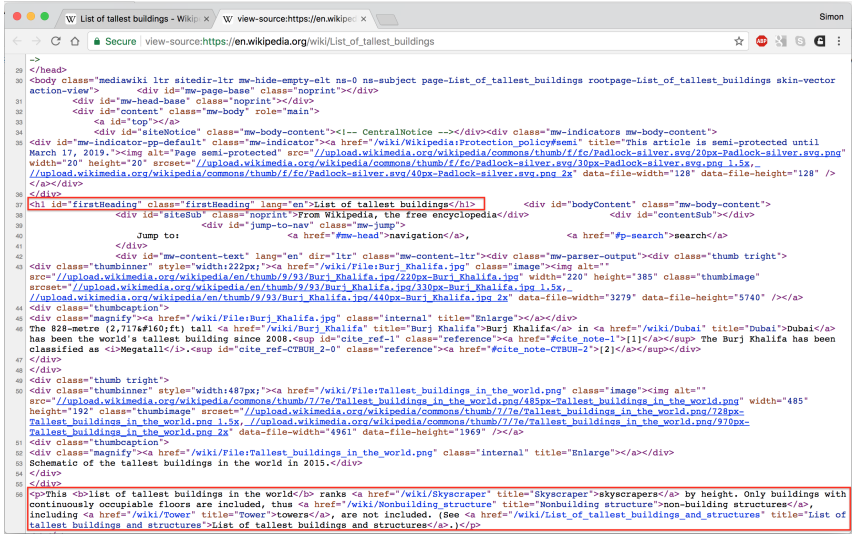
- HTML (and JavaScript) code can be ugly...



Inspecting the HTML source code

Example

- HTML (and JavaScript) code can be ugly...
- but looking more closely, we can find the displayed information

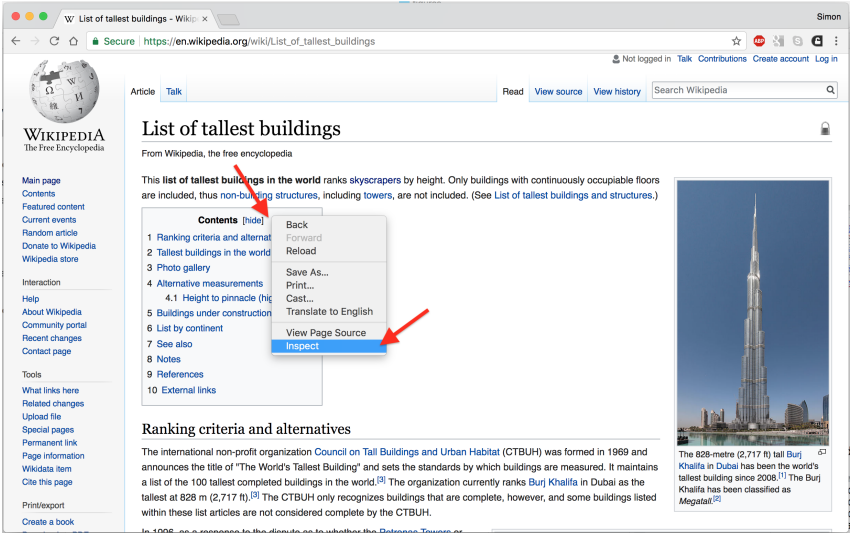


Inspecting the live HTML tree

Inspecting individual elements and the live HTML tree

Example

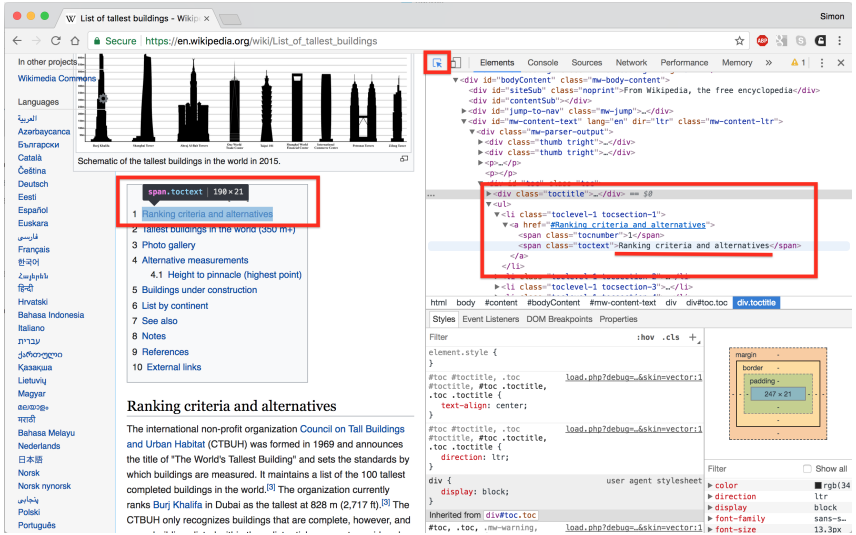
- another way of inspecting the HTML code is to use the **Web Developer Tools**
- to that end, do:
- right-click on element of interest
- select "Inspect"



Inspecting individual elements and the live HTML tree

Example

- the Web Developer Tools window pops up
- corresponding part in the HTML tree is highlighted
- you can hover over other parts of the code to get an instant view of the page at that position
- click on arrows to expand/hide tags



Summary

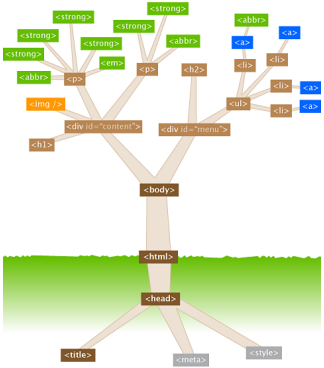
Summary

When to inspect the complete page source

- check whether data is in static source code
- for small HTML files: understand structure
- count tables

When to inspect individual elements using the Web Inspector Tools

- almost always
- partcularly useful to construct XPath/CSS selector expressions
- to monitor dynamic changes in the DOM tree (see later)



Source:
http://watershedcreative.com/naked/html-tree.html