Web Scraping with R

Alex Sanchez and Francesc Carmona
Genetics Microbiology and Statistics Department

Universitat de Barcelona

October 2022

Contents

1. Introduction to web technologies & Overview
2. The basics: HTML and CSS
3. Scraping web pages with R using `rvest`
4. XML, XPath and the DOM. Parsing XML
5. Using APIs to get data

Objectives and Competences

- Become familiar with technologies for content dissemination on the web.
- Information extraction from web-formatted data.
- Become familiar *-that is, know how to do it-* with the different tasks involved in web scraping.
- Learn how to set up and execute successful web scraping projects (making them as automatic, robust and error-resistant as possible).

We need data, and the web is full of it

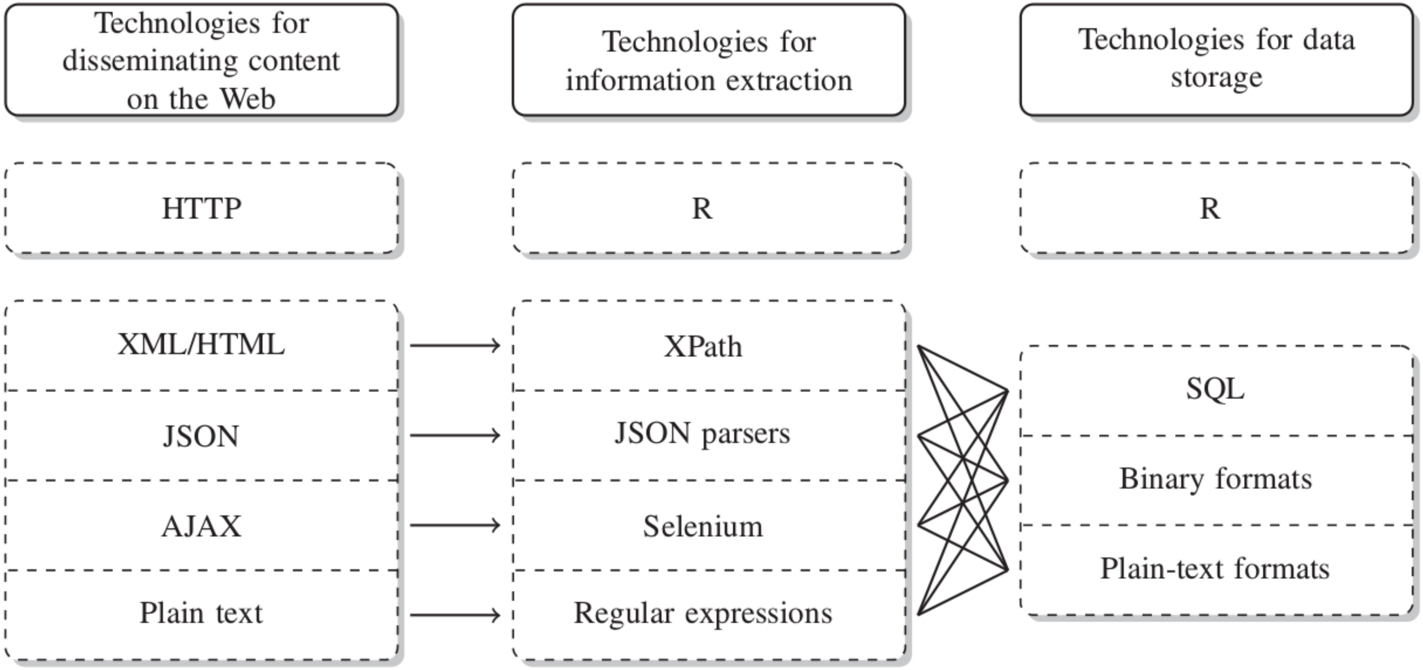
- Whatever our job is, it often relies on having the appropriate data to work with.
- The web has plenty of data
  - In 2008, an estimated 154 million HTML tables (out of the 14.1 billion) contain 'high quality relational data'!!!
  - Hard to quantify how much more exists outside of HTML Tables, but there is an estimate of at least 30 million lists with 'high quality relational data'.
- Accessing the data in the web is the topic of this course

What we need to know

- Technologies that allow the *distribution of content on the Web*.
- Techniques & Tools for *collecting* (as opposite to distributing) data from the web.
- In the way to acquiring these abilites we may learn many useful things that don't necessarily have to do with web scraping such as:
    - HTML/CSS for creating web -and non web- pages.
    - XML for sharing many types of data (also pdf, excel or epub)
    - Regular expressions for describing patterns in strings.
    - A variety of text mining and other interesting topics, such as "Sentiment Analysis" for analyzing data from Twitter, Linkedin etc.

Data Technologies

Technologies for disseminating, extracting, and storing web data.



Technologies for disseminating content on the Web

Technologies for information extraction

Technologies for data storage

HTTP

R

R

XML/HTML

JSON

AJAX

Plain text

XPath

JSON parsers

Selenium

Regular expressions

SQL

Binary formats

Plain-text formats

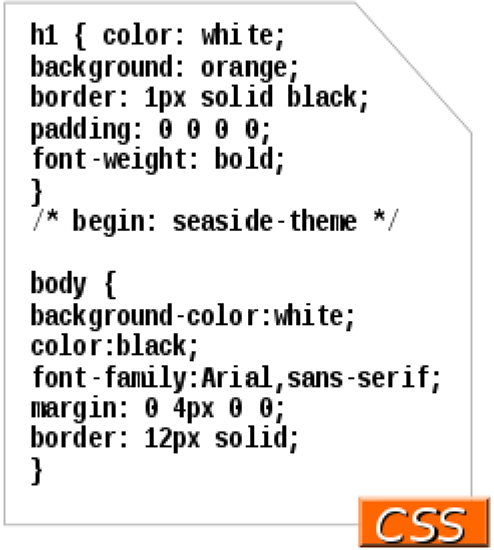
Source: Automated Data Collection with R

Technologies (1): HTML

```
<html>
<head id="html_head">
  <title>Website name</title>
    <noscript><meta http-equiv="X-Frame
<script type="text/javascript">
</script>
ageNow = new Date().getTime(
bar">              ground" ;
```

- **H**ypertext **M**arkup **L**anguage (HTML) is the language that all browsers understand.
- Not a dedicated data storage format but
- First option for containing information we look for.
- A minimum understanding of html required

Technologies (2): CSS



```
h1 { color: white;
background: orange;
border: 1px solid black;
padding: 0 0 0 0;
font-weight: bold;
}
/* begin: seaside-theme */

body {
background-color:white;
color:black;
font-family:Arial,sans-serif;
margin: 0 4px 0 0;
border: 12px solid;
}
```

- CSS is the language for describing the presentation of Web pages, including colors, layout, and fonts.
- It allows one to adapt the presentation to different types of devices, such as large screens, small screens, or printers.
- CSS is independent of HTML and can be used with any XML-based markup language.

Technologies (3): XML

```
<?xml version="1.0" encoding="UTF-8"?>
<foo>Hello World!</foo>
```

- E**X**tensible **M**arkup **L**anguage or XML is one of the most popular formats for exchanging data over the Web.
- XML is "just" data wrapped in user-defined tags.
- The user-defined tags **make XML much more flexible** for storing data than HTML.

Technologies (4): XPath

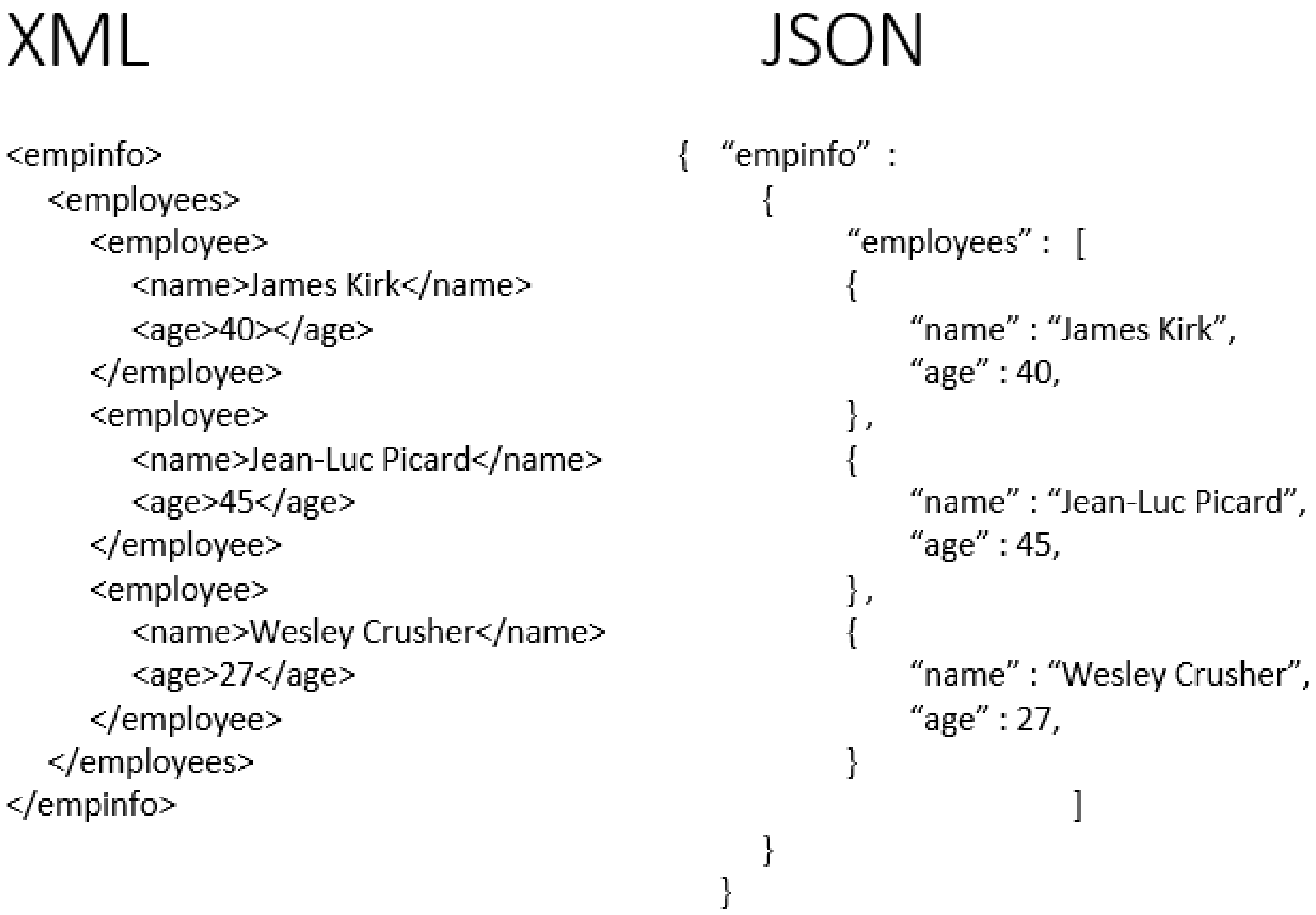
<xpath>

- The **X**ML **Path**Language provides a powerful syntax for handling specific elements of an XML document and, to the same extent, HTML web pages in a simple way.
- XML is "just" data wrapped in user-defined tags.
- The user-defined tags **make XML much more flexible** for storing data than HTML.

Technologies (4): JSON

- JavaScript Object Notation or JSON
- JSON is a lightweight data-interchange format
- JSON is language independent but uses javascript syntax
- JSON is a syntax for storing and exchanging data.
- JSON is an easier-to-use alternative to XML

Technologies (5) XML vs JSON



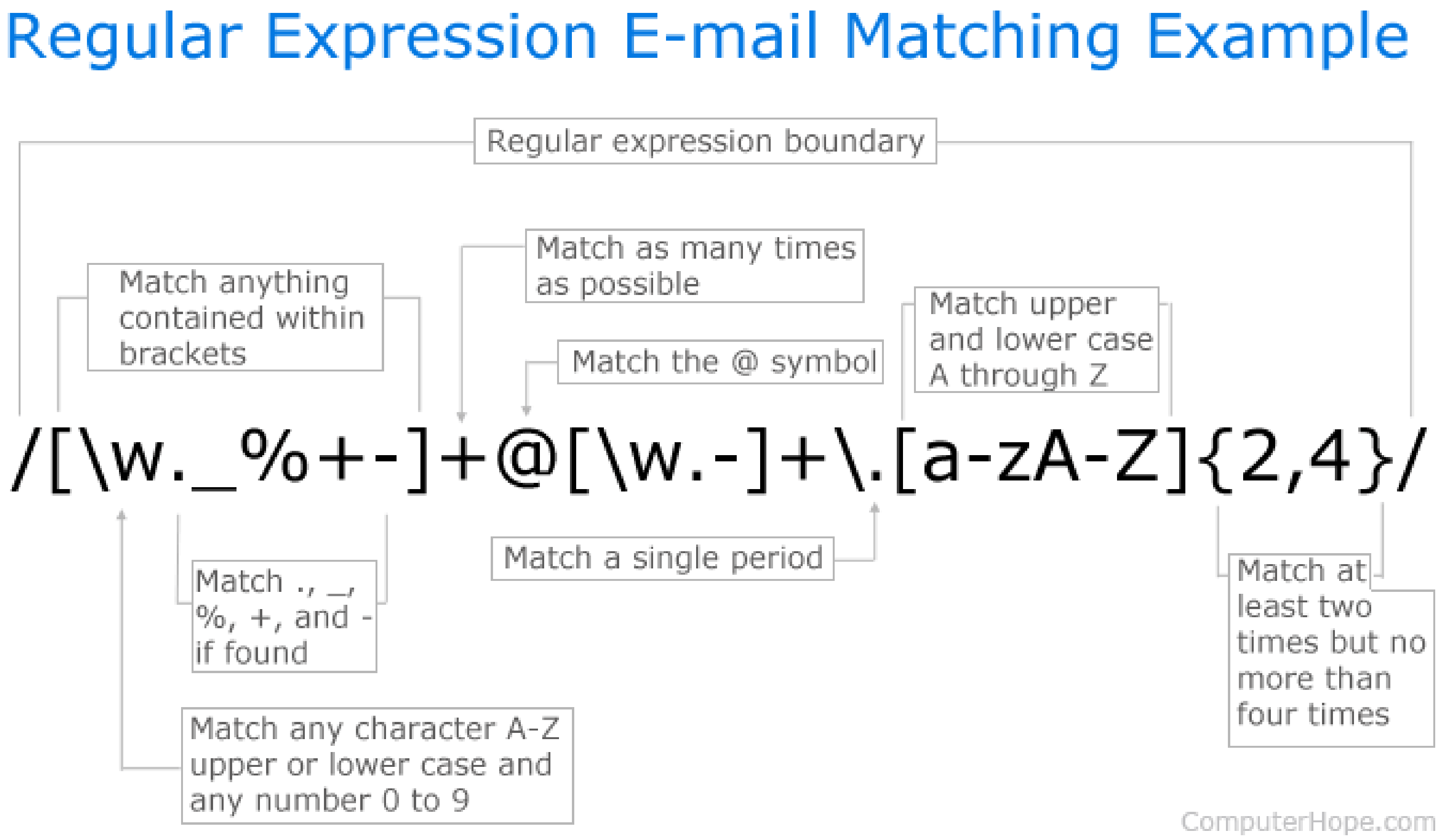
XML

```
<empinfo>
   <employees>
      <employee>
         <name>James Kirk</name>
         <age>40></age>
      </employee>
      <employee>
         <name>Jean-Luc Picard</name>
         <age>45</age>
      </employee>
      <employee>
         <name>Wesley Crusher</name>
         <age>27</age>
      </employee>
   </employees>
</empinfo>
```

JSON

```
{   “empinfo”  :
         {
               “employees” :   [
               {
                    “name” : “James Kirk”,
                    “age” : 40,
               } ,
               {
                    “name” : “Jean-Luc Picard”,
                    “age” : 45,
               } ,
               {
                    “name” : “Wesley Crusher”,
                    “age” : 27,
               }
                                   ]
         }
   }
```

Technologies (6): Regular Expressions



Regular Expression E-mail Matching Example

Regular expression boundary

Match anything contained within brackets

Match as many times as possible

Match the @ symbol

Match upper and lower case A through Z

```
/[\w._%+-]+@[\w.-]+\.[a-zA-Z]{2,4}/
```

Match ., _, %, +, and - if found

Match a single period

Match at least two times but no more than four times

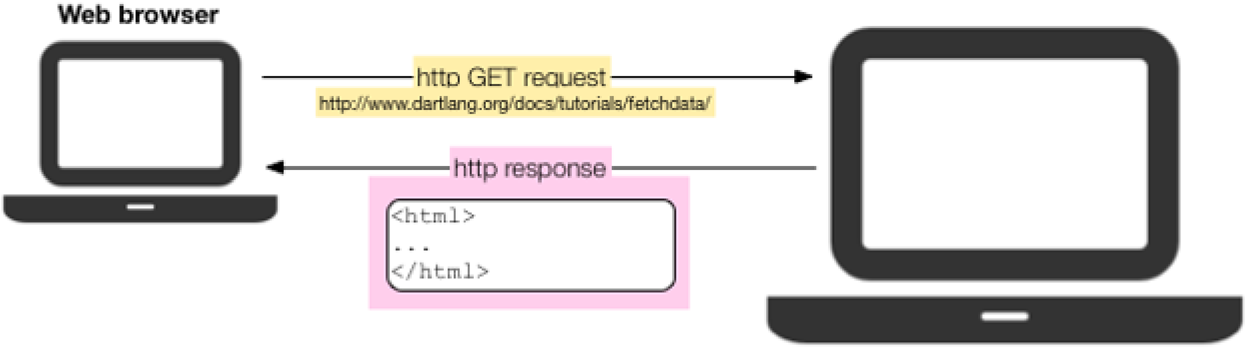
Match any character A-Z upper or lower case and any number 0 to 9

ComputerHope.com

So what is web scraping?

- Web scraping (web harvesting or web data extraction) is how we name computer software techniques for extracting information from websites.
  - See Wikipedia
- Web scraping focuses on the *transformation of unstructured data* on the web, typically in web format such as HTML, XML or JSON, into *structured* data that can be stored and analyzed in a central local database or spreadsheet.
  - Instead of structured data, if using R, we can think of *tidy* data.

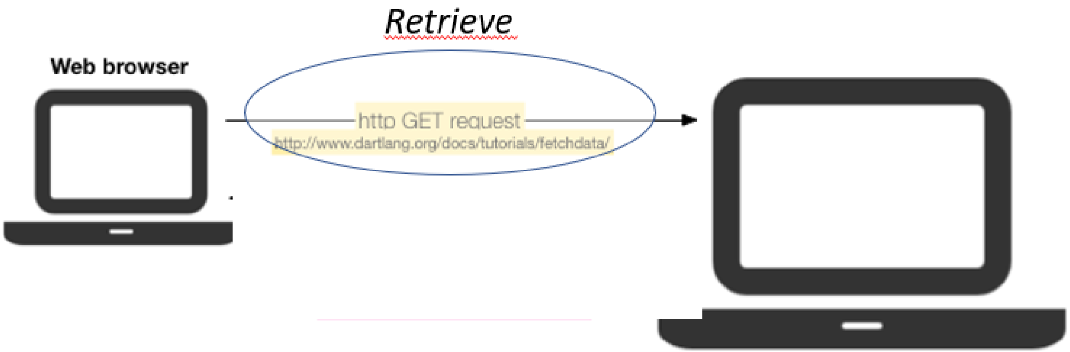
Understanding web communication: http



```
<html>
...
</html>
```

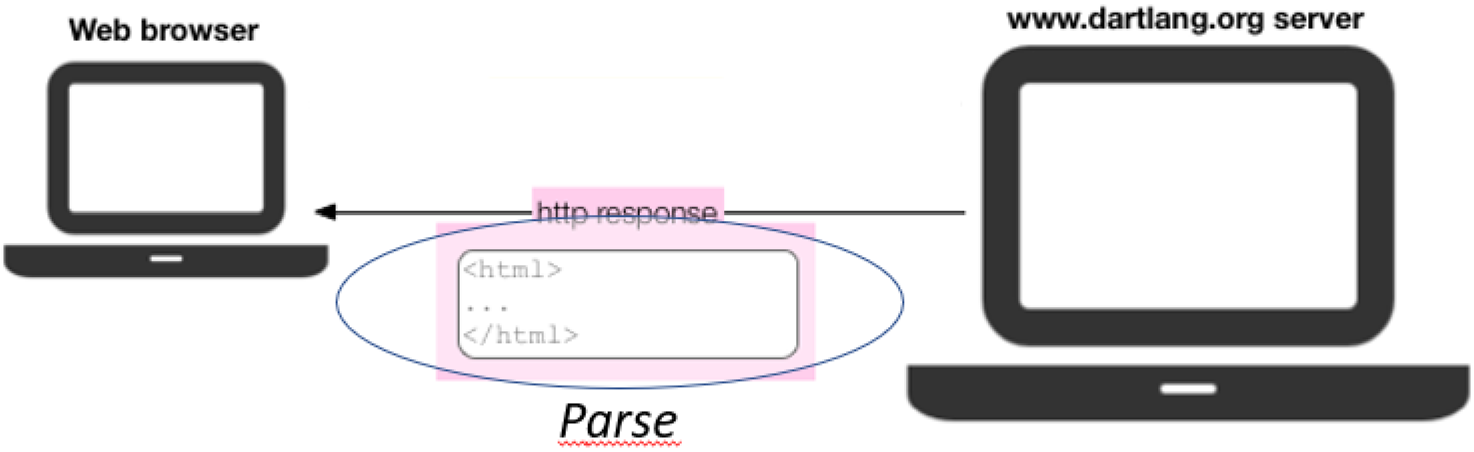
- User/Client asks for information: **http request**
- Server returns the information **http response**
- Data acquisition may be performed at two levels
  - Requesting information directly from the server
  - Parsing the response emited by the server

Requesting information directly



- Two ways for direct information retrieval:
    - in raw form through http GET requests
    - through an Application Programming Interface (API)
        - many APIs for retrieving data from "typical" places such as Twitter, Amazon, Linkedin, etc.
            - In R: "RLinkedin" "TwiteR" etc. packages
        - APIs require an authorization/user identification

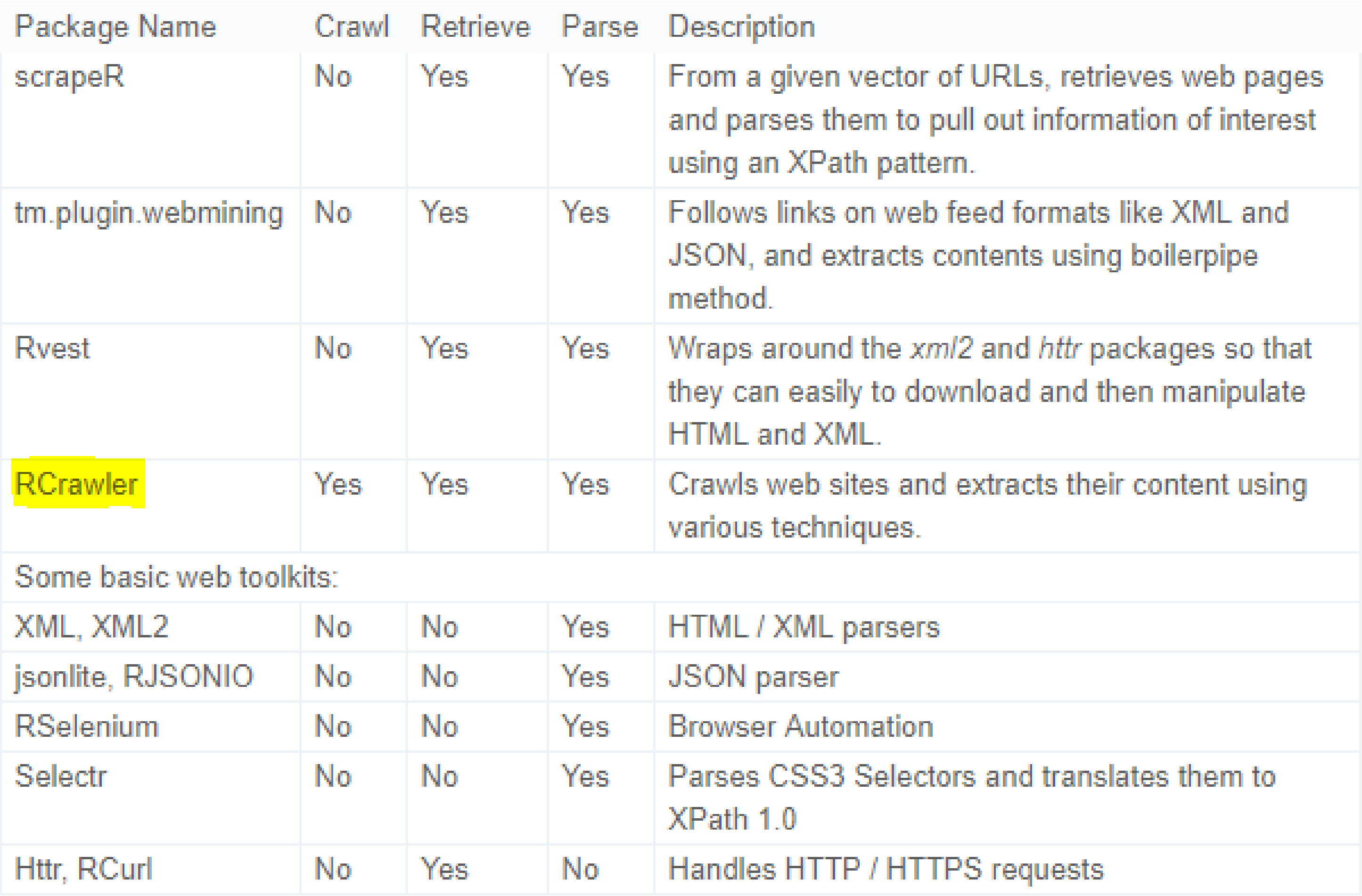
Parsing the server's response



- Parser tools extract information from the response sent by the server to the browser.
- The response is usually an HTML / XML document.
- Parsers exploit the hierarchical structure of HTML / XML to extract information and convert it into R objects
- R packages: `rvest`, `selectR`, `XML`, `xml2`

The R scraping toolkit

- Comparison of some popular R packages for data collection.

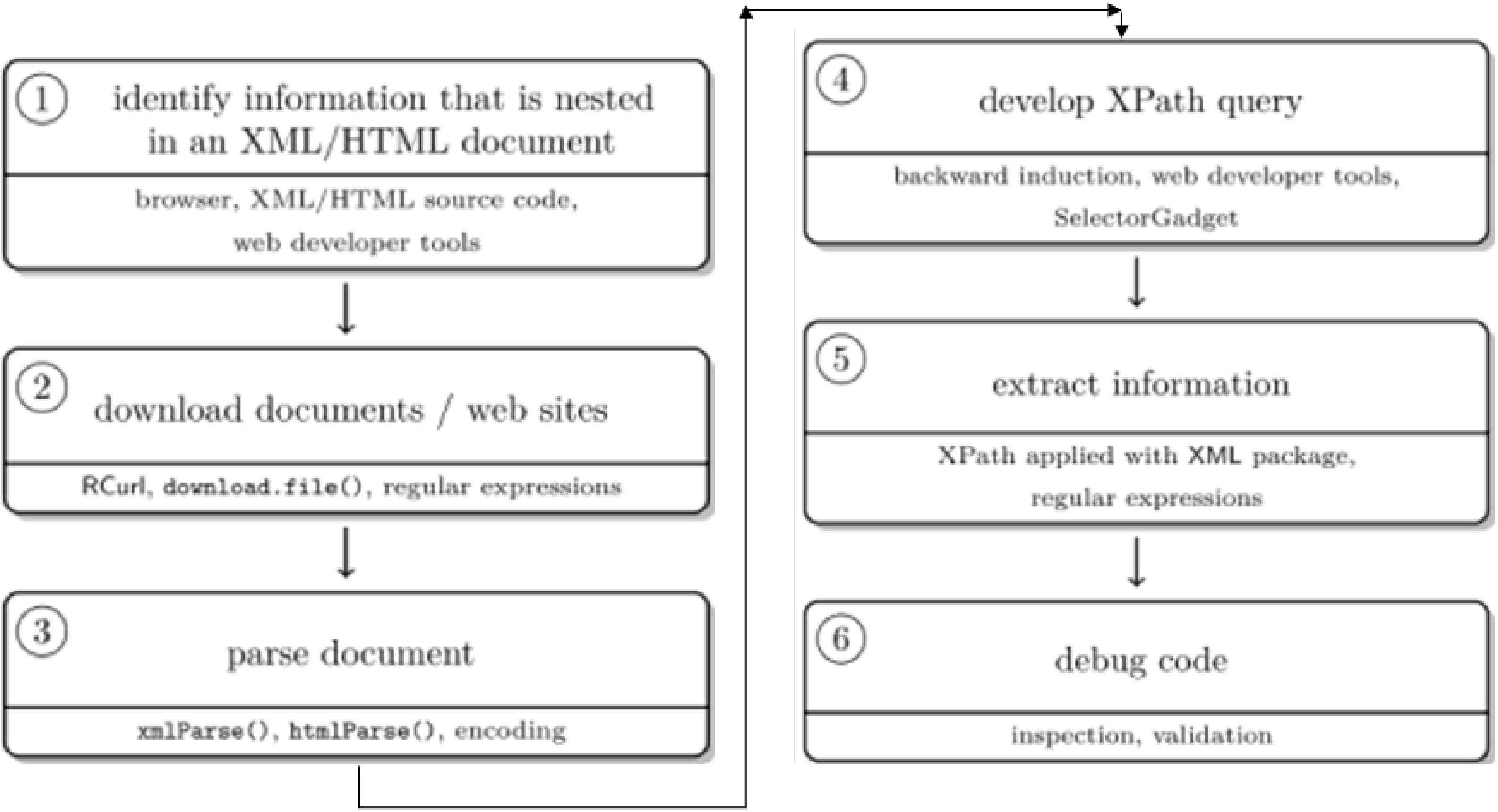


Package Name	Crawl	Retrieve	Parse	Description
scrapeR	No	Yes	Yes	From a given vector of URLs, retrieves web pages and parses them to pull out information of interest using an XPath pattern.
tm.plugin.webmining	No	Yes	Yes	Follows links on web feed formats like XML and JSON, and extracts contents using boilerpipe method.
Rvest	No	Yes	Yes	Wraps around the *xml2* and *httr* packages so that they can easily to download and then manipulate HTML and XML.
RCrawler	Yes	Yes	Yes	Crawls web sites and extracts their content using various techniques.
Some basic web toolkits:				
XML, XML2	No	No	Yes	HTML / XML parsers
jsonlite, RJSONIO	No	No	Yes	JSON parser
RSelenium	No	No	Yes	Browser Automation
Selectr	No	No	Yes	Parses CSS3 Selectors and translates them to XPath 1.0
Httr, RCurl	No	Yes	No	Handles HTTP / HTTPS requests

Web scraping and R

- Web scraping has been developed independently of R.
  See for example:
  - Scraping the Web for Arts and Humanities
  - Introduction to Web Scraping using Scrapy and Postgres
- There is a lot of information on scraping using python
- However if you feel comfortable working with R it is possible that you can absorbe easier and faster some of the the vast amount of resources for getting data from the web with R.

The scrapping process



1 identify information that is nested in an XML/HTML document

browser, XML/HTML source code, web developer tools

2 download documents / web sites

RCurl, download.file(), regular expressions

3 parse document

xmlParse(), htmlParse(), encoding

4 develop XPath query

backward induction, web developer tools, SelectorGadget

5 extract information

XPath applied with XML package, regular expressions

6 debug code

inspection, validation

Example: Heritage sites in danger

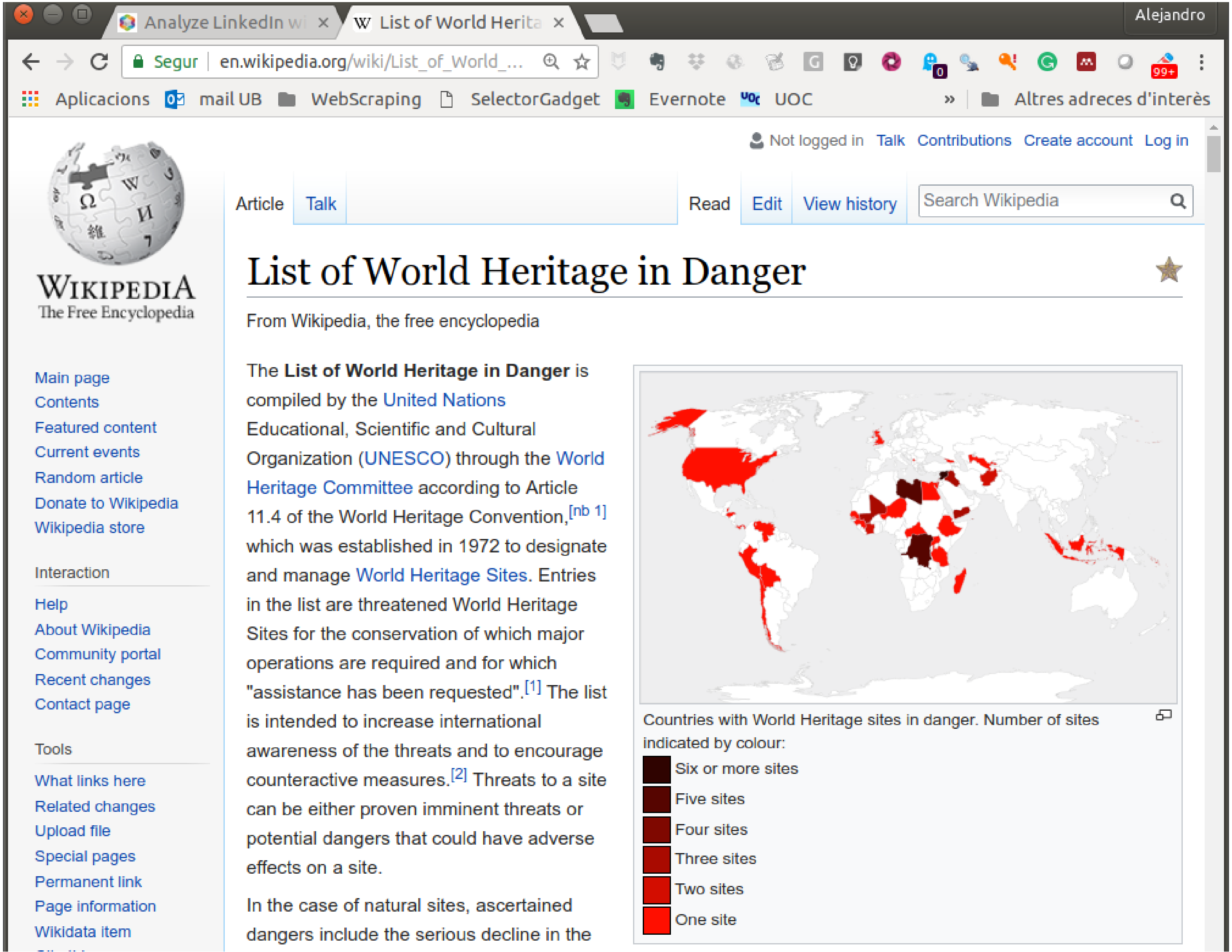
- The UNESCO is an organization of the United Nations which, among other things, fights for the preservation of the world's natural and cultural heritage.
- As November 2013 there are 981 heritage sites, most of which of are man-made like the Pyramids of Giza, but also natural phenomena like the Great Barrier Reef are listed.
- Unfortunately, some of the awarded places are threatened by human intervention.
- These are the questions that we want to examine in this first case study.
  - *Which sites are threatened and where are they located?*
  - *Are there regions in the world where sites are more endangered than in others?*
  - *What are the reasons that put a site at risk?*

Working through the case study with R

- This case study has been adapted from chapter 1 of the book Automated Data Collection with R (ADCR, from now on).
- Its goal is not to be exhaustive but providing a first example of a situation where we obtain and analyze data from the web.
- The goal is to tabulate and plot a list of endangered sites available in https://en.wikipedia.org/wiki/List_of_World_Heritage_in_Danger.
- We proceed as follows:

1. Go to the web and locate the desired information
2. Download the pages (here, HTML document)
3. Extract HTML table into an R object
4. Clean the data and build a data.frame
5. Plot and analyze

Example 1a: Wikipedia page



List of World Heritage in Danger

From Wikipedia, the free encyclopedia

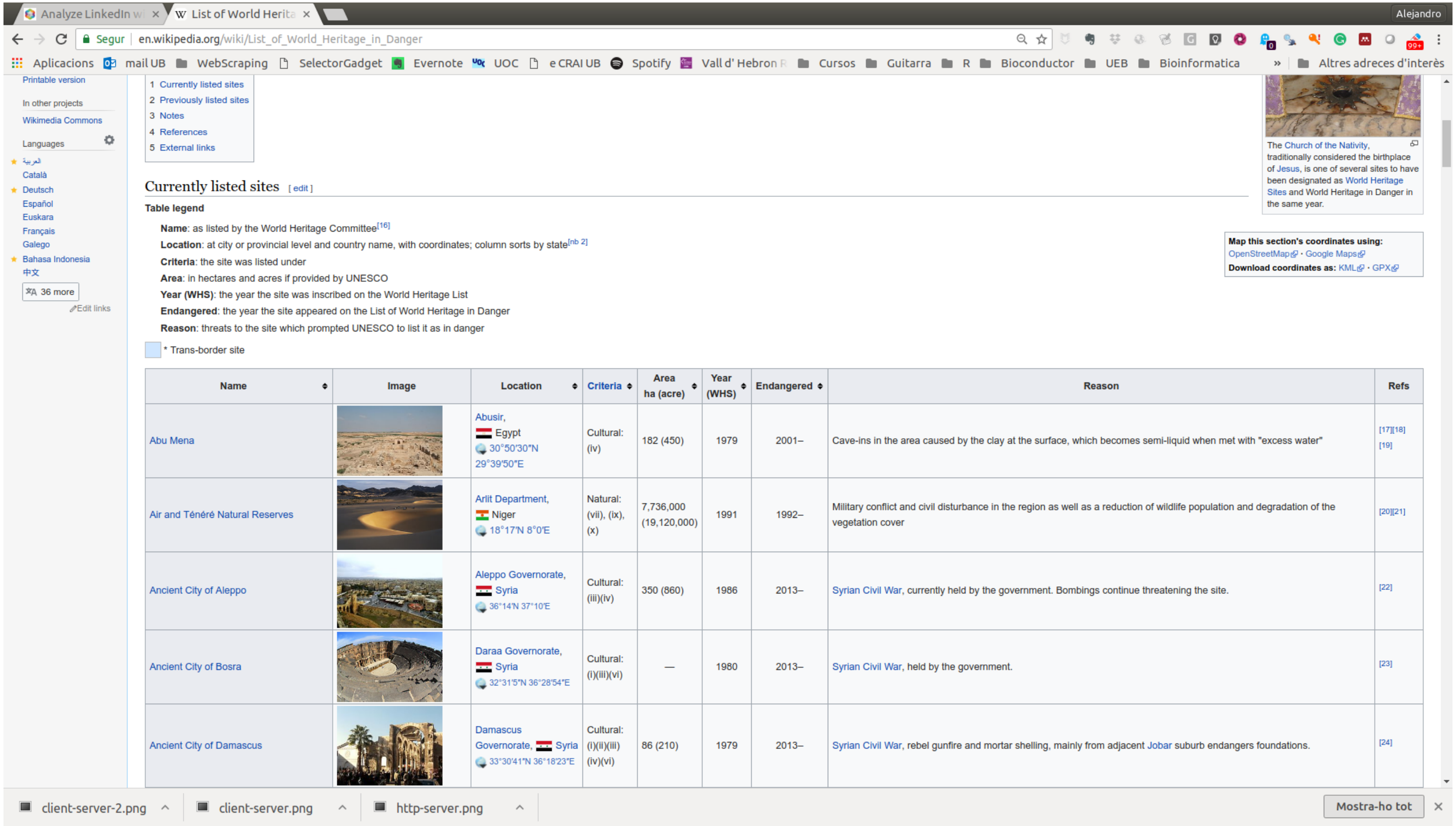
The **List of World Heritage in Danger** is compiled by the United Nations Educational, Scientific and Cultural Organization (UNESCO) through the World Heritage Committee according to Article 11.4 of the World Heritage Convention,[nb 1] which was established in 1972 to designate and manage World Heritage Sites. Entries in the list are threatened World Heritage Sites for the conservation of which major operations are required and for which "assistance has been requested".[1] The list is intended to increase international awareness of the threats and to encourage counteractive measures.[2] Threats to a site can be either proven imminent threats or potential dangers that could have adverse effects on a site.

In the case of natural sites, ascertained dangers include the serious decline in the

Countries with World Heritage sites in danger. Number of sites indicated by colour:

- ■ Six or more sites
- ■ Five sites
- ■ Four sites
- ■ Three sites
- ■ Two sites
- ■ One site

Example 1b: Locate desired table



1 Currently listed sites
2 Previously listed sites
3 Notes
4 References
5 External links

The Church of the Nativity, traditionally considered the birthplace of Jesus, is one of several sites to have been designated as World Heritage Sites and World Heritage in Danger in the same year.

Currently listed sites [edit]

Map this section's coordinates using: OpenStreetMap · Google Maps
Download coordinates as: KML · GPX

Table legend

Name: as listed by the World Heritage Committee[16]
Location: at city or provincial level and country name, with coordinates; column sorts by state[nb 2]
Criteria: the site was listed under
Area: in hectares and acres if provided by UNESCO
Year (WHS): the year the site was inscribed on the World Heritage List
Endangered: the year the site appeared on the List of World Heritage in Danger
Reason: threats to the site which prompted UNESCO to list it as in danger

* Trans-border site

Name	Image	Location	Criteria	Area ha (acre)	Year (WHS)	Endangered	Reason	Refs
Abu Mena		Abusir, Egypt 30°50′30″N 29°39′50″E	Cultural: (iv)	182 (450)	1979	2001–	Cave-ins in the area caused by the clay at the surface, which becomes semi-liquid when met with "excess water"	[17][18][19]
Air and Ténéré Natural Reserves		Arlit Department, Niger 18°17′N 8°0′E	Natural: (vii), (ix), (x)	7,736,000 (19,120,000)	1991	1992–	Military conflict and civil disturbance in the region as well as a reduction of wildlife population and degradation of the vegetation cover	[20][21]
Ancient City of Aleppo		Aleppo Governorate, Syria 36°14′N 37°10′E	Cultural: (iii)(iv)	350 (860)	1986	2013–	Syrian Civil War, currently held by the government. Bombings continue threatening the site.	[22]
Ancient City of Bosra		Daraa Governorate, Syria 32°31′5″N 36°28′54″E	Cultural: (i)(iii)(vi)	—	1980	2013–	Syrian Civil War, held by the government.	[23]
Ancient City of Damascus		Damascus Governorate, Syria 33°30′41″N 36°18′23″E	Cultural: (i)(ii)(iii)(iv)(vi)	86 (210)	1979	2013–	Syrian Civil War, rebel gunfire and mortar shelling, mainly from adjacent Jobar suburb endangers foundations.	[24]

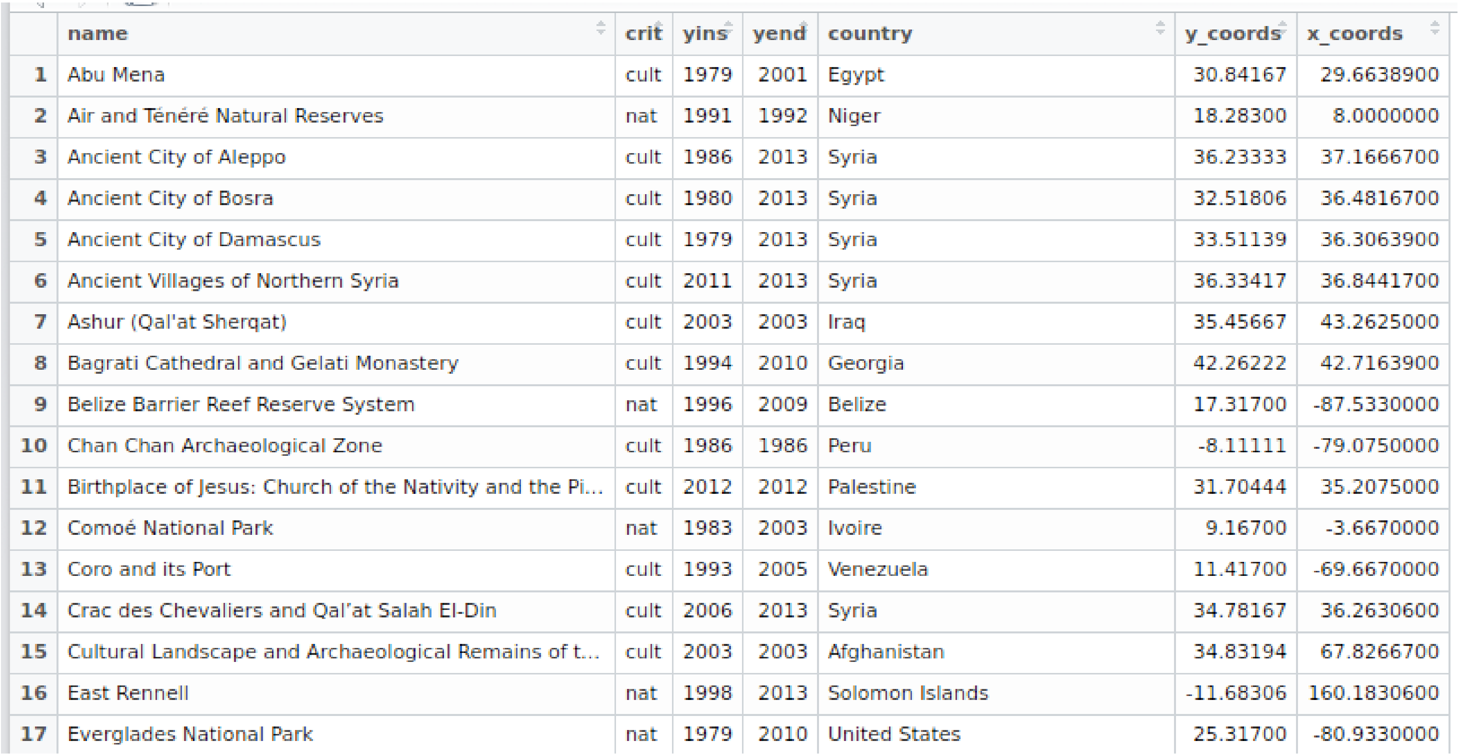
Example 1c: R code (1)

```
# load packages
library(stringr); library(XML); library(maps)
#--- parsing from locally stored HTML file
heritage_parsed ← htmlParse("worldheritagedanger.htm")
#--- Extract table from web page and select desired table
danger_table ← readHTMLTable(heritage_parsed, stringsAsFactors = FALSE, which =
danger_table ← danger_table[,c(1,3,4,6,7)]
colnames(danger_table) ← c("name","locn","crit","yins","yend")
#--- Clean data
danger_table$crit ← ifelse(str_detect(danger_table$crit, "Natural")T, "nat", "c
# cleanse years
danger_table$yins ← as.numeric(danger_table$yins)
danger_table$yend ← as.numeric(unlist(str_extract_all(danger_table$yend, "[[:di
#--- get countries
```

Example 1c: R code (2)

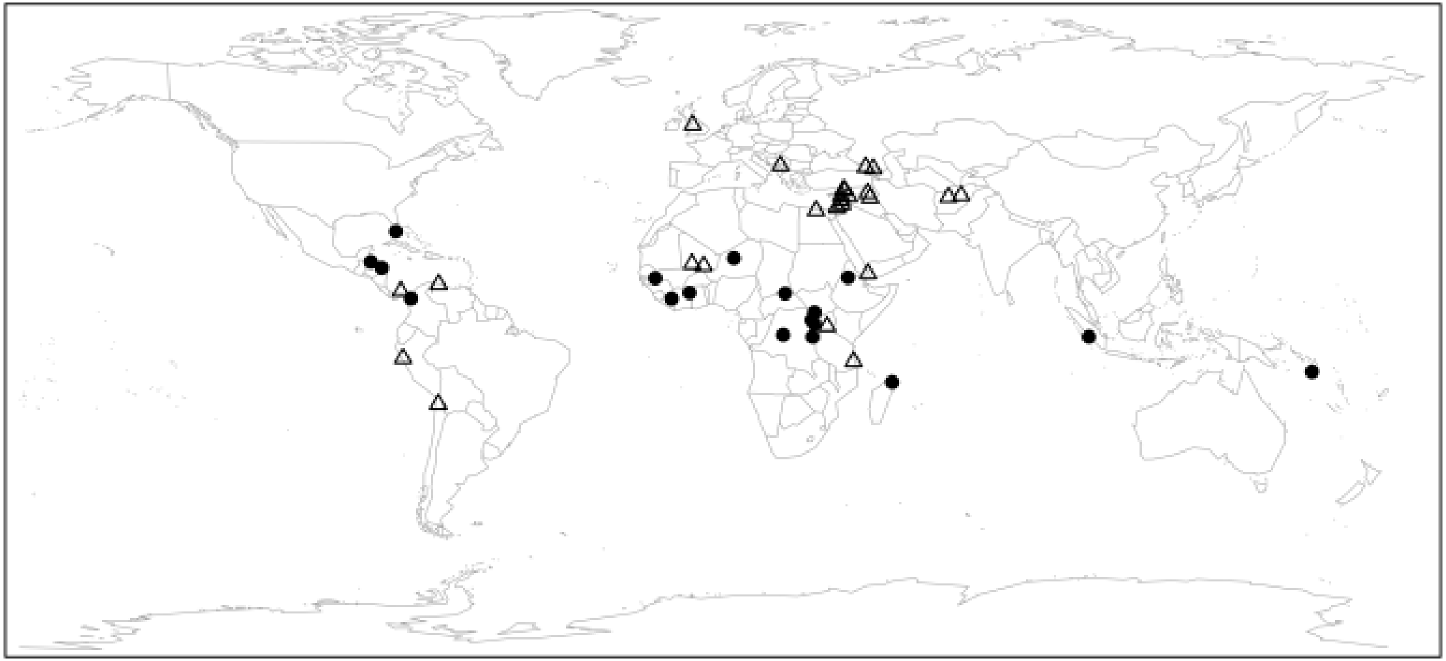
```r
#--- get countries
reg <- "[[:alpha:] ]+(?=[[:digit:]])"
danger_table$country <- str_extract(danger_table$locn , reg)
#--- get coordinates
reg_y <- "[/][ -]*[[:digit:]]*[.]*[[:digit:]]*[;]"
reg_x <- "[;][ -]*[[:digit:]]*[.]*[[:digit:]]*"
danger_table$y_coords <- as.numeric(str_sub(str_extract(danger_table$locn, reg_y
danger_table$x_coords <-  as.numeric(str_sub(str_extract(danger_table$locn, reg_
#--- plot endangered heritage sites
par(oma=c(0,0,0,0)); par(mar=c(0,0,0,0))
pch <- ifelse(danger_table$crit  "nat", 19, 2)
map("world", col = "darkgrey", lwd = .5, mar = c(0.1,0.1,0.1,0.1))
points(danger_table$x_coords, danger_table$y_coords, pch = pch, col = "black", c
box()
```

Example 1d: We have an R data frame



	name	crit	yins	yend	country	y_coords	x_coords
1	Abu Mena	cult	1979	2001	Egypt	30.84167	29.6638900
2	Air and Ténéré Natural Reserves	nat	1991	1992	Niger	18.28300	8.0000000
3	Ancient City of Aleppo	cult	1986	2013	Syria	36.23333	37.1666700
4	Ancient City of Bosra	cult	1980	2013	Syria	32.51806	36.4816700
5	Ancient City of Damascus	cult	1979	2013	Syria	33.51139	36.3063900
6	Ancient Villages of Northern Syria	cult	2011	2013	Syria	36.33417	36.8441700
7	Ashur (Qal'at Sherqat)	cult	2003	2003	Iraq	35.45667	43.2625000
8	Bagrati Cathedral and Gelati Monastery	cult	1994	2010	Georgia	42.26222	42.7163900
9	Belize Barrier Reef Reserve System	nat	1996	2009	Belize	17.31700	-87.5330000
10	Chan Chan Archaeological Zone	cult	1986	1986	Peru	-8.11111	-79.0750000
11	Birthplace of Jesus: Church of the Nativity and the Pi…	cult	2012	2012	Palestine	31.70444	35.2075000
12	Comoé National Park	nat	1983	2003	Ivoire	9.16700	-3.6670000
13	Coro and its Port	cult	1993	2005	Venezuela	11.41700	-69.6670000
14	Crac des Chevaliers and Qal’at Salah El-Din	cult	2006	2013	Syria	34.78167	36.2630600
15	Cultural Landscape and Archaeological Remains of t…	cult	2003	2003	Afghanistan	34.83194	67.8266700
16	East Rennell	nat	1998	2013	Solomon Islands	-11.68306	160.1830600
17	Everglades National Park	nat	1979	2010	United States	25.31700	-80.9330000

Example 1e: And now the plot



References and resources (1)

Books

- Automated Data Collection from the Web with R, by Munzer, Rubba, Meisner & Nyhulis. Wiley.
- XML and Web Technologies for Data Science with R. Deborah Nolan & Duncan Temple Lang. UseR!
- Introduction to Data Technologies. Duncan Murdoch.

Courses

- Datacamp: Web scraping in R
- Learn to scrape any website with R

References and resources (2)

Web documents/bookdown/etc.

- Introduction to Computing with Data, particularly part IX, Data Technologies
- Web scraping with R by Steve Pittard

Tutorials/Blog posts/etc.

- Getting Data from the Web with R, by Gaston Sánchez.
- Web scraping for the humanities and social sciences, Rolf Fredheim and Aiora Zabala.
- R-bloggers posts on *Web Scraping*
- And see also CRAN Web Services and Technologies task view