Web Data Collection with R

Session 3: Basic Web Technologies

Sascha Göbel
sascha.goebel@uni-konstanz.de

November 18th, 2020

Today: Web technologies

- URL components
- use of browser to locate information in source code of websites
- HTML and XML syntax
- parsing HTML and XML documents

HTML

Hypertext Markup Language

- plain text with markup
- standard for construction of websites
- instructs browser what, where, and how to display
- when we use a browser, we use HTML
- no need to write HTML, but we need to understand it

Introducing the running examples

(1) register of employees at the PolSci department of Uni KN

(2) travels of president Trump

(3) UK e-petitions

- let's have a look at the URLs and inspect the pages

HTML

Hyper**t**ext **M**arkup **L**anguage

- plain text with markup
- standard for construction of websites
- instructs browser what, where, and how to display
- when we use a browser, we use HTML
- no need to write HTML, but we need to understand it

Introducing the running examples

(1) register of employees at the PolSci department of Uni KN

(2) travels of president Trump

(3) UK e-petitions

- let's have a look at the URLs and inspect the pages

HTML Syntax

Tree structure

- inspecting HTML source code is a crucial step in web data collection
- HTML document is systematically and hierarchically structured
- elements must be strictly nested within each other

Tags

- structural function and enclose content
- enclosed by < and >
- start tag – <tag>
- end tag – </tag>
- element – <tag> ... </tag>

HTML Syntax

Tree structure

- inspecting HTML source code is a crucial step in web data collection
- HTML document is systematically and hierarchically structured
- elements must be strictly nested within each other

Tags

- structural function and enclose content
- enclosed by < and >
- start tag – <tag>
- end tag – </tag>
- element – <tag> ... </tag>

Attributes

- provide additional information about elements
- always specified in the start tag
- expressed as name-value pairs, name= "value"
- e.g., **href** in the <**a**> tag

Special characters

- reserved for control purposes
- need specific expression to display them literally

Attributes

- provide additional information about elements
- always specified in the start tag
- expressed as name-value pairs, name= "value"
- e.g., **href** in the <**a**> tag

Special characters

- reserved for control purposes
- need specific expression to display them literally

Table 2.1 HTML entities

Character	Entity number	Entity name	Explanation
"	&#34;	&quot;	quotation mark
'	&#39;	&apos;	apostrophe
&	&#38;	&amp;	ampersand
<	&#60;	&lt;	less than
>	&#62;	&gt;	greater than
			non-breaking space
§	&#167;	&sect;	section
Á	&#192;	&Agrave;	A with grave accent
É	&#200;	&Egrave;	E with grave accent
á	&#224;	&agrave;	a with grave accent
é	&#232;	&egrave;	e with grave accent
♡	&#9829;	&hearts;	heart
	&#66001;		plumed head (Phaistos Disc)

Note: For a more comprehensive list of HTML entities, visit http://unicode-table.com

Source: Munzert et al. 2015

Common tags

- <**a**> – anchor tag
- <**div**> and <**span**> – organizational tags
- <**p**> – paragraphs tag
- <**h1**>, <**h2**>, **...** – headline tags
- <**ul**>, <**ol**>, <**dl**> – list tags
- <**li**> – list item tag
- <**br**> – line break tag
- <**b**>, <**i**>, <**strong**> – emphasis/layout tags
- <**table**>, <**tr**>, <**td**>, <**th**> – table tags
- <**form**> – server interaction tag
- <**script**> – script container tag

Common attributes

- **href** – specifies URL of linked page
- **src** – specifies path to image
- **width** and **height** – specify width and heigth of image
- **style** – specifies color, font, size, etc.
- **id** – identifies a unique element
- **class** – specifies class name(s) of element, reference elements with CSS

Parsing

HTML in R

- simply reading in plain text files won't do
- need representation of HTML that R understands
- called the Document Object Model (DOM)
- a queryable data object
- using the xml2 package

XML

eXtensible Markup Language

- very similar to HTML, start tag-end tag logic
- elements contain data value
- elements can be closed in the start tag with a slash /
- data wrapped in user-defined tags
- attributes describe element in more detail
- hierarchical tree structure
- much more flexible for data storage than HTML
- though not very efficient