Web Data Collection with R

Xpath

Peter Meißner / 2016-02-29 – 2016-03-04 / ECPR WSMT

HTML/XML tree structure again

Running Example

How XPath works . . .

How CSS-Selectors Work . . .

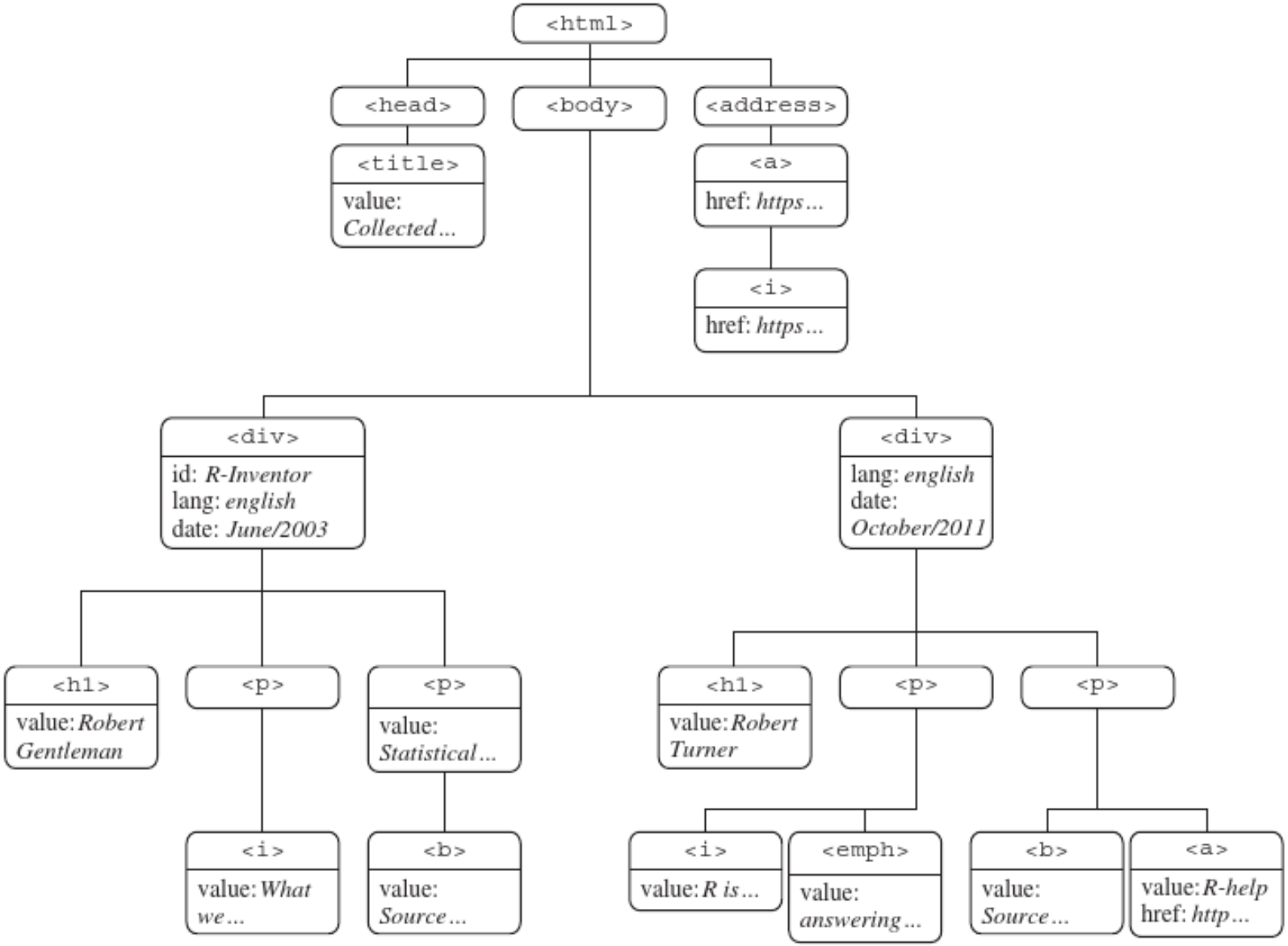
R-Packages and Functions

Selector Gadget and Developer Tools to the Rescue

HTML/XML tree structure again

HTML/XML tree structure, nodes and attributes

http://www.r-datacollection.com/materials/html/fortunes.html



Running Example

running example

```
require(rvest)
require(stringr)
```

```
url <-
"http://pmeissner.com/downloads/fortunes.html"
fname <- basename(url)

if(!file.exists(fname)){
  download.file(url, fname)
}

html <- read_html(fname)
```

running example

```
xml2::html_structure(html)
```

```
## <html>
##   {text}
##   <head>
##     {text}
##     <title>
##       {text}
##     {text}
##     <style>
##       {cdata}
##     {text}
##   {text}
##   <body>
##     {text}
##     <div#R_Inventor [lang, date]>
##       {text}
##       <h1.pink>
```

How XPath works . . .

XPath? What is it all about?

- XPath is a query language for XML (Extensible Markup Language) documents
- XML examples are: XML, HTML, SVG, GML, KML, EPUB, RSS, Office Open XML, OpenDocument
- in XPath on selects nodes describing the paths that lead to that path

How XPath Works . . .

- builds on
  - **hierarchy** (select parent, child, sibling, . . . node)
  - **node names** (select node by name)
  - **node values** (select node by value)
  - **attribute name and value** (select node on attribute value)
  - **further functions** (select depending on more complex derivates of the above)
    - e.g. name, string_length, contains, count, position, . . .
  - **operators**
    - e.g. `|`, `+`, `-`, `=`, `!=`, `<=`, `or`, `and`, . . .
- allows to extract
  - node values
  - attribute values
- . . . from single nodes and node sets

explicit path

```
x="/html/body/div[2]/h1"
html_nodes(html, xpath=x)
```

```
## {xml_nodeset (1)}
## [1] <h1 class="pink">Rolf Turner</h1>
```

path anywhere in hierarchy

```
html_nodes(html, xpath="//h1")
```

```
## {xml_nodeset (2)}
## [1] <h1 class="pink">Robert Gentleman</h1>
## [2] <h1 class="pink">Rolf Turner</h1>
```

path anywhere in hierarchy / attribute

```
html_nodes(html, xpath="//a/@href")
```

```
## {xml_nodeset (2)}
## [1]  href="https://stat.ethz.ch/mailman/listinfo/r-help"
## [2]  href="www.r-datacollectionbook.com"
```

path anywhere in hierarchy / function

```
html_nodes(html, xpath="//p/i/text()")
```

```
## {xml_nodeset (2)}
## [1] 'What we have is nice, but we need something very di
## [2] 'R is wonderful, but it cannot work magic'
```

path anywhere in hierarchy / indexing

```
html_nodes(html, xpath="//div[1]/p/i")
```

```
## {xml_nodeset (1)}
## [1] <i>'What we have is nice, but we need something very
```

path anywhere in hierarchy / indexing

```
html_nodes(html, xpath="//div")
```

```
## {xml_nodeset (2)}
## [1] <div id="R_Inventor" lang="english" date="June/2003"
## [2] <div lang="english" date="October/2011">\n\t\t\t<h1
```

```
html_nodes(html, xpath="//div[1]")
```

```
## {xml_nodeset (1)}
## [1] <div id="R_Inventor" lang="english" date="June/2003"
```

```
html_nodes(html, xpath="//div[1]/p/i/text()")
```

```
## {xml_nodeset (1)}
## [1] 'What we have is nice, but we need something very di
```

node / attribute contains/is equal . . .

```
html_nodes(html, xpath="//div[@date='October/2011']")
```

```
## {xml_nodeset (1)}
## [1] <div lang="english" date="October/2011">\n\t\t\t<h1
```

```
html_nodes(html, xpath="//div[contains(@date, 'October/201
```

```
## {xml_nodeset (1)}
## [1] <div lang="english" date="October/2011">\n\t\t\t<h1
```

```
html_nodes(html, xpath="//div[contains(.//a/@href, 'https')
```

```
## {xml_nodeset (1)}
## [1] <div lang="english" date="October/2011">\n\t\t\t<h1
```

node / attribute contains/is equal . . .

```
html_nodes(html, xpath="//a[contains(@href, 'https')]")
```

```
## {xml_nodeset (1)}
## [1] <a href="https://stat.ethz.ch/mailman/listinfo/r-he
```

```
html_nodes(html, xpath="//a[contains(., 'homepage')]")
```

```
## {xml_nodeset (1)}
## [1] <a href="www.r-datacollectionbook.com">\n\t\t\t\t<i
```

node parent

```
html_nodes(html, xpath="//a")
```

```
## {xml_nodeset (2)}
## [1] <a href="https://stat.ethz.ch/mailman/listinfo/r-hel
## [2] <a href="www.r-datacollectionbook.com">\n\t\t\t\t<i>
```

```
html_nodes(html, xpath="//a/..")
```

```
## {xml_nodeset (2)}
## [1] <p>\n\t\t\t\t<b class="pink">Source: </b>\n\t\t\t\t<
## [2] <address>\n\t\t\t<a href="www.r-datacollectionbook.c
```

all nodes everywhere

```
html_nodes(html, xpath="//*")
```

```
## {xml_nodeset (23)}
##  [1] <html> \n\t<head>\n\t\t<title>Collected R wisdoms</
##  [2] <head>\n\t\t<title>Collected R wisdoms</title>\n\t\
##  [3] <title>Collected R wisdoms</title>
##  [4] <style><![CDATA[\n\t\t\t.pink    {color:pink;}\n\t\
##  [5] <body>\n\t\t<div id="R_Inventor" lang="english" dat
##  [6] <div id="R_Inventor" lang="english" date="June/2003
##  [7] <h1 class="pink">Robert Gentleman</h1>
##  [8] <p>\n\t\t\t\t\t<i>'What we have is nice, but we need
##  [9] <i>'What we have is nice, but we need something ver
## [10] <p>\n\t\t\t\t\t<b class="pink">Source: </b>Statistica
## [11] <b class="pink">Source: </b>
## [12] <div lang="english" date="October/2011">\n\t\t\t\t<h1
## [13] <h1 class="pink">Rolf Turner</h1>
## [14] <p>\n\t\t\t\t\t<i>'R is wonderful, but it cannot work
## [15] <i>'R is wonderful, but it cannot work magic'</i>
```

all nodes' text everywhere

```
html_nodes(html, xpath="//*/text()")
```

```
## {xml_nodeset (43)}
##  [1]  \n\t
##  [2] \n\t\t
##  [3] Collected R wisdoms
##  [4] \n\t\t
##  [5] <![CDATA[\n\t\t\t.pink    {color:pink;}\n\t\t]]>
##  [6] \n\t
##  [7] \n\t
##  [8] \n\t\t
##  [9] \n\t\t\t
## [10] Robert Gentleman
## [11] \n\t\t\t
## [12] \n\t\t\t\t
## [13] 'What we have is nice, but we need something very c
## [14] \n\t\t\t
## [15] \n\t\t\t
```

that node or the other

```
html_nodes(html, xpath="//i | //b")
```

```
## {xml_nodeset (5)}
## [1] <i>'What we have is nice, but we need something very
## [2] <b class="pink">Source: </b>
## [3] <i>'R is wonderful, but it cannot work magic'</i>
## [4] <b class="pink">Source: </b>
## [5] <i>The book homepage</i>
```

using axis :: parent

```
html_nodes(html, xpath="//a/..")
```

```
## {xml_nodeset (2)}
## [1] <p>\n\t\t\t\t<b class="pink">Source: </b>\n\t\t\t\t<
## [2] <address>\n\t\t\t<a href="www.r-datacollectionbook.c
```

```
html_nodes(html, xpath="//a/parent::*")
```

```
## {xml_nodeset (2)}
## [1] <p>\n\t\t\t\t<b class="pink">Source: </b>\n\t\t\t\t<
## [2] <address>\n\t\t\t<a href="www.r-datacollectionbook.c
```

```
html_nodes(html, xpath="//a/parent::p")
```

```
## {xml_nodeset (1)}
## [1] <p>\n\t\t\t\t<b class="pink">Source: </b>\n\t\t\t\t<
```

using axis :: child

```
html_nodes(html, xpath="//p/i")
```

```
## {xml_nodeset (2)}
## [1] <i>'What we have is nice, but we need something very
## [2] <i>'R is wonderful, but it cannot work magic'</i>
```

```
html_nodes(html, xpath="//p/child::*")
```

```
## {xml_nodeset (7)}
## [1] <i>'What we have is nice, but we need something very
## [2] <b class="pink">Source: </b>
## [3] <i>'R is wonderful, but it cannot work magic'</i>
## [4] <br/>
## [5] <emph>answering a request for automatic generation o
## [6] <b class="pink">Source: </b>
## [7] <a href="https://stat.ethz.ch/mailman/listinfo/r-hel
```

```
html_nodes(html, xpath="//p/child::i")
```

using axis :: ancestor

```
html_nodes(html, xpath="//b/ancestor::*")
```

```
## {xml_nodeset (6)}
## [1] <html> \n\t<head>\n\t\t<title>Collected R wisdoms</t
## [2] <body>\n\t\t<div id="R_Inventor" lang="english" date
## [3] <div id="R_Inventor" lang="english" date="June/2003"
## [4] <p>\n\t\t\t\t<b class="pink">Source: </b>Statistical
## [5] <div lang="english" date="October/2011">\n\t\t\t<h1
## [6] <p>\n\t\t\t\t<b class="pink">Source: </b>\n\t\t\t\t<
```

```
html_nodes(html, xpath="//b/ancestor::*/text()")
```

```
## {xml_nodeset (20)}
##  [1]  \n\t
##  [2] \n\t
##  [3] \n\t\t
##  [4] \n\t\t\t
##  [5] \n\t\t\t
```

using axis :: descendant

```
html_nodes(html, xpath="//p/descendant::*")
```

```
## {xml_nodeset (7)}
## [1] <i>'What we have is nice, but we need something very
## [2] <b class="pink">Source: </b>
## [3] <i>'R is wonderful, but it cannot work magic'</i>
## [4] <br/>
## [5] <emph>answering a request for automatic generation o
## [6] <b class="pink">Source: </b>
## [7] <a href="https://stat.ethz.ch/mailman/listinfo/r-hel
```

```
html_nodes(html, xpath="//p/descendant::*/text()")
```

```
## {xml_nodeset (6)}
## [1] 'What we have is nice, but we need something very di
## [2] Source:
## [3] 'R is wonderful, but it cannot work magic'
## [4] answering a request for automatic generation of 'dat
```

using axis :: following-sibling / preceding-sibling

```
html_nodes(html, xpath="//b/..")
```

```
## {xml_nodeset (2)}
## [1] <p>\n\t\t\t\t<b class="pink">Source: </b>Statistical
## [2] <p>\n\t\t\t\t<b class="pink">Source: </b>\n\t\t\t\t<
```

```
html_nodes(html, xpath="//b/following-sibling::*")
```

```
## {xml_nodeset (1)}
## [1] <a href="https://stat.ethz.ch/mailman/listinfo/r-hel
```

How CSS-Selectors Work . . .

How CSS-Selectors Work . . .

- CSS-Selectors were designed to apply Styles to HTML elements
- While XPath is build around the idea of hierarchy and tree-structure first and foremost meaning that paths lead to data, with CSS-S selection is more set-like.
- CSS-S is used and written for Web-Designers so it might be less-powerful-complete-systematic than XPath but it is also less intimidating and easier to write
- selection on class and id attributes is super easy
  - **name** (select nodes by name)
  - **id** (select node id attribute)
  - **node values** (select node by value)
  - **attribute name and value** (select node on attribute value)
  - **hierarchy** (select depending on the position in path)

selecting nodes by name

```
html_nodes(html, "p")
```

```
## {xml_nodeset (4)}
## [1] <p>\n\t\t\t\t<i>'What we have is nice, but we need s
## [2] <p>\n\t\t\t\t<b class="pink">Source: </b>Statistical
## [3] <p>\n\t\t\t\t<i>'R is wonderful, but it cannot work
## [4] <p>\n\t\t\t\t<b class="pink">Source: </b>\n\t\t\t\t<
```

```
html_nodes(html, "b, i")
```

```
## {xml_nodeset (5)}
## [1] <i>'What we have is nice, but we need something very
## [2] <b class="pink">Source: </b>
## [3] <i>'R is wonderful, but it cannot work magic'</i>
## [4] <b class="pink">Source: </b>
## [5] <i>The book homepage</i>
```

selecting nodes by class

```
html_nodes(html, ".pink")
```

```
## {xml_nodeset (4)}
## [1] <h1 class="pink">Robert Gentleman</h1>
## [2] <b class="pink">Source: </b>
## [3] <h1 class="pink">Rolf Turner</h1>
## [4] <b class="pink">Source: </b>
```

selecting nodes by id

```
html_nodes(html, css = "#R_Inventor")
```

```
## {xml_nodeset (1)}
## [1] <div id="R_Inventor" lang="english" date="June/2003'
```

```
html_nodes(html, css = "[id='R_Inventor']")
```

```
## {xml_nodeset (1)}
## [1] <div id="R_Inventor" lang="english" date="June/2003'
```

selecting nodes by attribute

```
html_nodes(html, css = "[lang]")
```

```
## {xml_nodeset (2)}
## [1] <div id="R_Inventor" lang="english" date="June/2003"
## [2] <div lang="english" date="October/2011">\n\t\t\t<h1
```

```
html_nodes(html, css = "[href]")
```

```
## {xml_nodeset (2)}
## [1] <a href="https://stat.ethz.ch/mailman/listinfo/r-hel
## [2] <a href="www.r-datacollectionbook.com">\n\t\t\t\t<i>
```

selecting nodes by attribute value

```
html_nodes(html, css = "[id=R_Inventor]") # equal
```

```
## {xml_nodeset (1)}
## [1] <div id="R_Inventor" lang="english" date="June/2003'
```

```
html_nodes(html, css = "[id^=R]")          # starts
```

```
## {xml_nodeset (1)}
## [1] <div id="R_Inventor" lang="english" date="June/2003'
```

```
html_nodes(html, css = "[id$=r]")          # ends
```

```
## {xml_nodeset (1)}
## [1] <div id="R_Inventor" lang="english" date="June/2003'
```

```
html_nodes(html, css = "[id*=ven]")        # conatains
```

```
## {xml_nodeset (1)}
## [1] <div id="R_Inventor" lang="english" date="June/2003'
```

selecting nodes by path characteristics : decendant

```
html_nodes(html, css = "i")
```

```
## {xml_nodeset (3)}
## [1] <i>'What we have is nice, but we need something very
## [2] <i>'R is wonderful, but it cannot work magic'</i>
## [3] <i>The book homepage</i>
```

```
html_nodes(html, css = "a i")
```

```
## {xml_nodeset (1)}
## [1] <i>The book homepage</i>
```

selecting nodes by path characteristics :parent

```
html_nodes(html, css = "p > i")
```

```
## {xml_nodeset (2)}
## [1] <i>'What we have is nice, but we need something very
## [2] <i>'R is wonderful, but it cannot work magic'</i>
```

```
html_nodes(html, css = "a > i")
```

```
## {xml_nodeset (1)}
## [1] <i>The book homepage</i>
```

selecting nodes by path characteristics : first of type

```
html_nodes(html, css = "p:first-of-type")
```

```
## {xml_nodeset (2)}
## [1] <p>\n\t\t\t\t<i>'What we have is nice, but we need s
## [2] <p>\n\t\t\t\t<i>'R is wonderful, but it cannot work
```

selecting nodes by path characteristics : nth child of parent

```
html_nodes(html, css = "a:nth-child(1)")
```

```
## {xml_nodeset (1)}
## [1] <a href="www.r-datacollectionbook.com">\n\t\t\t\t<i>
```

```
html_nodes(html, css = "a:nth-child(2)")
```

```
## {xml_nodeset (1)}
## [1] <a href="https://stat.ethz.ch/mailman/listinfo/r-hel
```

selecting nodes by path characteristics : nth child of parent

```
html_nodes(html, css = "a:nth-last-child(1)")
```

```
## {xml_nodeset (2)}
## [1] <a href="https://stat.ethz.ch/mailman/listinfo/r-hel
## [2] <a href="www.r-datacollectionbook.com">\n\t\t\t\t<i>
```

```
html_nodes(html, css = "a:nth-last-child(2)")
```

```
## {xml_nodeset (0)}
```

selecting nodes by path characteristics : nth child of parent

```
html_nodes(html, css = "p:nth-of-type(1)")
```

```
## {xml_nodeset (2)}
## [1] <p>\n\t\t\t\t<i>'What we have is nice, but we need s
## [2] <p>\n\t\t\t\t<i>'R is wonderful, but it cannot work
```

R-Packages and Functions

rvest and XML

rvest (httr + xml2 + selectr)

- scraping centered package (download and extraction)
- HTML / XML
- XPath / CSS-S
- very handy and slick
- we use this

XML (xml)

- XML centered package (parsing and extraction)
- XPath
- much more powerful in terms of parsing (also SAX for LARGE documents)
- goes back to 1999 (according to README; you know just after the internet became a thing)
- two good sources cover that one: Nolan & Temple-Lang (2013): *XML and Web Technologies for Data Sciences with R*; Munzert et al (2014): *Automated Data Collection with R*

rvest's (important) XML handling functions

function	description
`read_html()`	parse HTML (file); all others based on
`html_node()`	extract a node via XPath and CSS-S.
`html_nodes()`	extract nodes via XPath and CSS-S.
`html_attr()`	get specific attribute value (node)
`html_attrs()`	get all attributes (node)
`html_text()`	get node's and children's text (node)
`xml_path()`	gives back the explicit path to nodes (node)
`html_structure()`	shows the structure of an HTML (doc)
`as_list()`	transform parsed XML / HTML to list (doc)
`html_children()`	get children of node (doc, node)
`xml_length()`	number of children (node)
`xml_parent()`	gives back parent of node (node)
`xml_parents()`	gives back all ancestors of node (node)
`xml_siblings()`	gives back nodes with the same parent (node)
`xml_type()`	gives back type (node, doc)

Selector Gadget and Developer Tools to the Rescue

Selector Gadget and Developer Tools to the Rescue

- building Xpath (CSS-S) expressions is an art (practice hard and be creative)
- . . . and easily and quickly becomes mind buggling and complicated . . .
- . . . there are however some tools that might help lessen the burden:
  - selectorgadget : http://selectorgadget.com/
  - developer tools :
    - Chrome: https://developer.chrome.com/devtools
    - Firefox: https://developer.mozilla.org/de/docs/Tools
    - Opera: http://www.opera.com/dragonfly/
    - Safari: https://developer.apple.com/safari/tools/
    - Edge: https://dev.windows.com/en-us/microsoft-edge/platform/documentation/f12-devtools-guide/