Title  Assemble Data Frames from HTML Tables
Version  0.7.1
Description  HTML tables are a valuable data source but extracting and recasting these data into a useful format can be tedious. This package allows to collect structured information from HTML tables. It is similar to readHTMLTable() of the XML package but provides three major advantages. First, the function automatically expands row and column spans in the header and body cells. Second, users are given more control over the identification of header and body rows which will end up in the R table, including semantic header information that appear throughout the body. Third, the function preprocesses table code, corrects common types of malformations, removes unneeded parts and so helps to alleviate the need for tedious post-processing.

Depends  R (>= 3.0.0)
Imports  XML (>= 3.98.1.3), httr (>= 1.0.0)
License  MIT + file LICENSE
LazyData  true
Suggests  testthat, knitr, tidyr
URL  https://github.com/crubba/htmltab
BugReports  https://github.com/crubba/htmltab/issues
VignetteBuilder  knitr
RoxygenNote  5.0.1
NeedsCompilation  no
Author  Christian Rubba [aut, cre]
Maintainer  Christian Rubba <christian.rubba@gmail.com>
Repository  CRAN
Date/Publication  2016-12-29 01:06:12
check_type

R topics documented:

- check_type .................................................. 2
- create_inbody .................................................. 3
- eval_body ....................................................... 3
- eval_header .................................................... 4
- get_body_xpath ................................................ 4
- get_cell_element .............................................. 5
- get_header_elements ........................................... 5
- get_head_xpath ................................................ 6
- get_span ......................................................... 6
- get_trindex ...................................................... 7
- htmltab ........................................................ 7
- identify_elements .............................................. 10
- normalize_tr ................................................... 10
- num_xpath ....................................................... 11
- rm_empty_cols .................................................. 11
- rm_empty_rows .................................................. 12
- rm_nuisance ..................................................... 12
- select_tab ...................................................... 13

Index

check_type

<table>
<thead>
<tr>
<th>check_type</th>
<th>Produce the table node</th>
</tr>
</thead>
</table>

Description

Produce the table node

Usage

check_type(doc, which, ...)

Arguments

doc the HTML document which can be a file name or a URL or an already parsed document (by XML's parsing functions)

which a vector of length one for identification of the table in the document. Either a numeric vector for the tables’ rank or a character vector that describes an XPath for the table

... additional arguments passed to htmlParse

Value

a table node
create_inbody

Reshape in table header information into wide format

Description
Reshape in table header information into wide format

Usage
create_inbody(tab, table.Node, trindex, xpath)

Arguments
- tab: the table data frame
- table.Node: the table node
- trindex: the tr index of the inbody rows
- xpath: the xpath for the inbody rows

Value
the modified R data frame

eval_body
Evaluate and deparse the body argument

Description
Evaluate and deparse the body argument

Usage
eval_body(arg)

Arguments
- arg: the body argument
---

**eval_header**

*Evaluate and deparse the header argument*

**Description**

Evaluate and deparse the header argument

**Usage**

`eval_header(arg)`

**Arguments**

- `arg` the header information

**Value**

evaluated header info

---

**get_body_xpath**

*Return body xpath*

**Description**

Return body xpath

**Usage**

`get_body_xpath(body, table.Node)`

**Arguments**

- `body` an information for the body rows
- `table.Node` the table node

**Value**

a character vector of XPath statements
**get_cell_element**

*Extracts cells elements*

**Description**
Extracts cells elements

**Usage**

```r
get_cell_element(cells, tag = "td | th", elFun, rm_escape, rm_whitespace)
```

**Arguments**

- **cells**: a list of cell nodes
- **tag**: a character vector that provides information used in the XPath expression to extract the correct elements
- **elFun**: a function that is executed over the header/body cell nodes
- **rm_escape**: a character vector that, if specified, is used to replace escape sequences in header and body cells (default value `' `)
- **rm_whitespace**: logical, should leading/trailing whitespace be removed from cell values (default value TRUE)?

**Value**

the body element

---

**get_header_elements**

*Extracts header elements*

**Description**
Extracts header elements

**Usage**

```r
get_header_elements(cells, tag = "td | th")
```

**Arguments**

- **cells**: a list of cell nodes
- **tag**: a character vector that provides information used in the XPath expression to extract the correct elements

**Value**

A list of header information from the cells
get_head_xpath  

Description

Return header xpath

Usage

get_head_xpath(header, table.Node)

Arguments

header  an information for the header rows
        table.Node  the table node

Value

a character vector of XPath statements

get_span  

Description

Extracts rowspan information

Usage

get_span(cells, span, tag = "td | th")

Arguments

cells  a list of cell nodes
        span  a character for the span element name
        tag  a character vector that provides information used in the XPath expression to extract the correct elements

Value

A list of row information from the cells
get_trindex

Return trindex given an XPath

Description

Return trindex given an XPath

Usage

get_trindex(xpath, table.Node)

Arguments

xpath XPath
table.Node the table node

htmltab

Assemble a data frame from HTML table data

Description

Robust and flexible methods for extracting structured information out of HTML tables

Usage

htmltab(doc, which = NULL, header = NULL, headerFun = function(node) XML::xmlValue(node), headerSep = "\n", body = NULL, bodyFun = function(node) XML::xmlValue(node), complementary = TRUE, fillNA = NA, rm_superscript = TRUE, rm_escape = "\n", rm_footnotes = TRUE, rm_nodata_cols = TRUE, rm_nodata_rows = TRUE, rm_invisible = TRUE, rm_whitespace = TRUE, colNames = NULL, ...)

Arguments

doc the HTML document which can be a file name or a URL or an already parsed document (by XML’s parsing functions)
which a vector of length one for identification of the table in the document. Either a numeric vector for the tables’ rank or a character vector that describes an XPath for the table
header the header formula, see details for specifics
headerFun a function that is executed over the header cell nodes
headerSep a character vector that is used as a separator in the construction of the table’s variable names (default ‘ » ‘)
body  a vector that specifies which table rows should be used as body information. A numeric vector can be specified where each element corresponds to a table row. A character vector may be specified that describes an XPath for the body rows. If left unspecified, htmltab tries to use semantic information from the HTML code

bodyFun a function that is executed over the body cell nodes

complementary logical, should htmltab ensure complementarity of header, inbody header and body elements (default TRUE)?

fillna character vector of symbols that are replaced by NA (default c(""))

rm_superscript logical, should superscript information be removed from header and body cells (default TRUE)?

rm_escape a character vector that, if specified, is used to replace escape sequences in header and body cells (default ‘’)

rm_footnotes logical, should semantic footer information be removed (default TRUE)?

rm_nodata_cols logical, should columns that have no alphanumeric data be removed (default TRUE)?

rm_nodata_rows logical, should rows that have no alphanumeric data be removed (default TRUE)?

rm_invisible logical, should nodes that are not visible be removed (default TRUE)? This includes elements with class ‘sortkey’ and ‘display:none’ style.

rm_whitespace logical, should leading/trailing whitespace be removed from cell values (default TRUE)?

colNames a character vector of column names, or a function that can be used to replace specific column names (default NULL)

... additional arguments passed to HTML parsers

Details

The header formula has the following format: level1 + level2 + level3 + ... level1 specifies the main header dimension (column names). This information must be for rows. level2 and deeper signify header dimensions that appear throughout the body. Those information must be for cell elements, not rows. Header information may be one of the following types:

- the NULL value (default). No information passed, htmltab will try to identify header elements through heuristics (heuristics only work for the main header)
- A numeric vector that retrieves rows in the respective position
- A character string of an XPath expression
- A function that when evaluated produces a numeric or character vector
- 0, when the process of finding the main header should be skipped (only works for main header)

Value

An R data frame
Author(s)
Christian Rubba \(<\text{http://www.christianrubba.com}\>\)

References
\(\text{https://github.com/crubba/htmltab}\)

Examples

```r
## Not run:
# When no spans are present, htmltab produces output close to XML's readHTMLTable(),
# but it removes many types of non-data elements (footnotes, non-visible HTML elements, etc)

url <- "http://en.wikipedia.org/wiki/World_population"
xp <- "//caption[starts-with(text(),'World historical')]//ancestor::table"
htmltab(doc = url, which = xp)

popFun <- function(node) {
  x <- XML::xmlValue(node)
  gsub('"', '', x)
}

htmltab(doc = url, which = xp, bodyFun = popFun)

# This table lacks header information. We provide them through colNames.
# We also need to set header = 0 to indicate that no header is present.
doc <- "http://en.wikipedia.org/wiki/FC_Bayern_Munich"
xp2 <- "//td[text()='Head coach']//ancestor::table"
htmltab(doc = doc, which = xp2, header = 0, encoding = "UTF-8", colNames = c("name", "role"))

# htmltab recognizes column spans and produces a one-dimension vector of variable information,
# also removes automatically superscript information since these are usually not of use.

xp3 <- "//table[7]"
bFun <- function(node) {
  x <- XML::xmlValue(node)
  gsub('%%', '', x)
}

htmltab(doc = doc, which = xp3, bodyFun = bFun)

htmltab("https://en.wikipedia.org/wiki/Arjen_Robben", which = 3,
header = 1:2)

# When header information appear throughout the body, you can specify their
# position in the header formula

htmltab(url, which = "//table[@id='team_game_logs']", header = . + "/td/.strong")
```
## identify_elements

Assemble XPath expressions for header and body

### Description

Assemble XPath expressions for header and body

### Usage

```r
identify_elements(table.Node, header, body, complementary = T)
```

### Arguments

- **table.Node**: the table node
- **header**: a vector that contains information for the identification of the header row(s). A numeric vector can be specified where each element corresponds to the table rows. A character vector may be specified that describes an XPath for the header rows. If left unspecified, htmltable tries to use semantic information from the HTML code
- **body**: a vector that specifies which table rows should be used as body information. A numeric vector can be specified where each element corresponds to a table row. A character vector may be specified that describes an XPath for the body rows. If left unspecified, htmltable tries to use semantic information from the HTML code
- **complementary**: logical, should htmltab ensure complementarity of header, inbody header and body elements (default TRUE)?

### Value

a character vector of XPath statements

## normalize_tr

Normalizes rows to be nested in tr tags, header in thead, body in tbody and numbers them

### Description

Normalizes rows to be nested in tr tags, header in thead, body in tbody and numbers them

### Usage

```r
normalize_tr(table.Node)
```
num_xpath

Arguments

  table.Node  the table node

Value

  the revised table node

num_xpath: Generate numeric XPath expression

Description

  Generate numeric XPath expression

Usage

  num_xpath(data)

Arguments

  data  the header XPath

rm_empty_cols

Remove columns which do not have data values

Description

  Remove columns which do not have data values

Usage

  rm_empty_cols(df, header)

Arguments

  df  a data frame
  header  the header vector

Value

  a data frame

See Also

  rm_nuisance, rm_empty_rows
**rm_empty_rows**

Remove rows which do not have data values

**Description**

Remove rows which do not have data values

**Usage**

\[ \text{rm_empty_rows}(\text{df}) \]

**Arguments**

- \( \text{df} \) a data frame

**Value**

a data frame

**See Also**

- \text{rm_nuisance}, \text{rm_empty_cols}

---

**rm_nuisance**

Remove nuisance elements from the the table code

**Description**

Remove nuisance elements from the the table code

**Usage**

\[ \text{rm_nuisance}(\text{table.Node}, \text{rm_superscript}, \text{rm_footnotes}, \text{rm_invisible}) \]

**Arguments**

- \text{table.Node} the table node
- \text{rm_superscript} logical, denotes whether superscript information should be removed from header and body cells (default value TRUE)
- \text{rm_footnotes} logical, denotes whether semantic footer information should be removed (default value TRUE)
- \text{rm_invisible} logical, should nodes that are not visible (display:none attribute) be removed?

**Value**

The revised table node
**select_tab**

**See Also**

`rm_empty_cols`

---

**select_tab**    *Selects the table from the HTML Code*

---

**Description**

Selects the table from the HTML Code

**Usage**

```
select_tab(which, Node)
```

**Arguments**

- `which`  
  a vector of length one for identification of the table in the document. Either a numeric vector for the tables’ rank or a character vector that describes an XPath for the table

- `Node`  
  the table node

- `...`  
  additional arguments passed to `htmlParse`

**Value**

a table node
Index

check_type, 2
create_inbody, 3

eval_body, 3
eval_header, 4

get_body_xpath, 4
get_cell_element, 5
get_head_xpath, 6
get_header_elements, 5
get_span, 6
get_trindex, 7

htmltab, 7

identify_elements, 10

normalize_tr, 10

num_xpath, 11

rm_empty_cols, 11, 12, 13
rm_empty_rows, 11, 12
rm_nuisance, 11, 12, 12

select_tab, 13