

Package ‘xmlr’

July 21, 2025

Version 0.1.2

Date 2020-05-06

Title Read, Write and Work with 'XML' Data

Maintainer Per Nyfelt <per@alipsa.se>

Depends R (>= 3.1.0)

Encoding UTF-8

Description 'XML' package for creating and reading and manipulating 'XML', with an object model based on 'Reference Classes'.

License MIT + file LICENSE

URL <https://github.com/Alipsa/xmlr>

BugReports <https://github.com/Alipsa/xmlr/issues>

Imports methods

Suggests testthat, knitr, rmarkdown

Collate 'xmlr.R' 'utils.R' 'AbstractClass.R' 'Content.R' 'Document.R'
'Text.R' 'Element.R' 'Stack.R' 'DomBuilder.R' 'Parser.R'
'xmlImporter.R' 'xmlConverter.R'

RoxygenNote 7.1.0

VignetteBuilder knitr

NeedsCompilation no

Author Per Nyfelt [cre, aut],
Alipsa HB [cph],
Steven Brandt [ctb]

Repository CRAN

Date/Publication 2020-05-12 09:30:02 UTC

Contents

AbstractClass-class	2
Content-class	2

Document-class	2
DomBuilder-class	3
Element-class	4
isRc	5
Parser-class	6
Stack-class	6
Text-class	6
xmlImporter	7
xmlr	8
xmlrToDataFrame	8
Index	10

AbstractClass-class	<i>Reference Class representing a non instantiable class</i>
---------------------	--

Description

An abstract base class with some utility methods

Content-class	<i>An abstract reference class representing content that can belong to an Element</i>
---------------	---

Description

#' @field m_parent the parent (if any)

Document-class	<i>Reference Class representing an XML document</i>
----------------	---

Description

The base container for the DOM

Usage

```
## S4 method for signature 'Document'
as.vector(x)

## S4 method for signature 'Document'
as.character(x)
```

Arguments

x the object to convert

Details

Methods allow access to the root element as well as the DocType and other document-level information.

Methods (by generic)

- as.vector: as.vector(Document)
- as.character: as.character(Document)

Methods

getBaseURI() return the URI from which this document was loaded

setBaseURI(uri) Sets the effective URI from which this document was loaded

DomBuilder-class	<i>Create a xmlr object tree based on parsing events</i>
------------------	--

Description

Create a xmlr object tree based on parsing events

Methods

endDocument() Event signalling parsing has completed

endElement(name) end element event; @param name the element name

startDocument() Event signalling parsing has begun

startElement(name, attributes) start element event; @param name the element name, @param attributes a named list of attributes

text(text) text event; @param text the character content of the Text node

Element-class

Element, A reference class representing an XML tag

Description

An XML element. Methods allow the user to get and manipulate its child elements and content, directly access the element's textual content, and manipulate its attributes.

Usage

```
## S4 method for signature 'Element'
as.vector(x)
```

```
## S4 method for signature 'Element'
as.character(x)
```

Arguments

x the object to convert

Methods (by generic)

- as.vector: as.vector(Element)
- as.character: as.character(Element)

Fields

name The local name of the element

contentList all the children of this element

attributeList a list of all the attributes belonging to this element

Methods

addAttributes(attributes) Add the supplied attributes to the attributeList of this Element

addContent(content) Appends the child to the end of the content list. return the parent (the calling object)

contentIndex(content) Find the position of the content in the contentList or -1 if not found

getAttribute(name) Get an attribute value

getAttributes() Get the list of attributes

getChild(name) Return the first child element matching the name

getChildren() Get all the child Elements belong to this Element

getContent() Returns the full content of the element as a List that may contain objects of type Text, Element, Comment, ProcessingInstruction, CDATA, and EntityRef

getName() Return the name of this Element

`getText()` Return the text content of this element if any
`hasAttributes()` return TRUE if this element has any attributes, otherwise FALSE
`hasChildren()` Return TRUE if this element has any child Element nodes
`hasContent()` return TRUE if this element has any content, otherwise FALSE
`hasText()` Return TRUE if this element has a Text node
`removeContent(content)` Remove the specified content from this element
`removeContentAt(index)` Remove the content at the given index and return the content that was removed
`setAttribute(name, value)` Add or replace an attribute, parameters will be converted to characters
`setAttributes(attributes)` Replace the attributes with this named list, NULL or empty list will remove all attributes, all values will be converted to characters
`setName(name)` Set the name of this Element
`setText(text)` Replace all content with the text supplied

 isRc

Common utility functions

Description

Common utility functions

Usage

```
isRc(x, clazz = "refClass")
```

Arguments

<code>x</code>	the object to check
<code>clazz</code>	the name of the class e.g. "Element" for the Element class. Optional, if omitted it checks that the object is a reference class

Value

A boolean indicating whether the object `x` belongs to the class or not

Functions

- `isRc`: Check if the object is a reference class, similar to `isS4()`.

Parser-class	<i>Parse an xml string and create sax like events</i>
--------------	---

Description

an XML parser based on an article on creating a quick and dirty xml parser by Steven Brandt:
<https://www.javaworld.com/article/2077493/java-tip-128-create-a-quick-and-dirty-xml-parser.html>

Stack-class	<i>A general purpose linked stack</i>
-------------	---------------------------------------

Description

A general purpose linked stack

Fields

size the size of the stack (number of elements in the stack)
 stackNode an environment containing the current element and the one under

Methods

peek() Get the top element from the stack without changing it
 pop() Pull the top element from the stack removing it from the stack
 push(val) Add an element to the top of the stack
 size() Get the current size of the stack

Text-class	<i>Reference class representing text content</i>
------------	--

Description

Reference class representing text content
 as.vector for Text classes
 as.character for Text classes

Usage

```
## S4 method for signature 'Text'
as.vector(x)

## S4 method for signature 'Text'
as.character(x)
```

Arguments

x the object to convert

Details

An XML character sequence. Provides a modular, parentable method of representing text.

Methods (by generic)

- `as.vector: as.vector(Text)`
- `as.character: as.character(Text)`

xmlImporter

XML import functions

Description

XML import functions

Usage

`parse.xmlstring(xml)`

`parse.xmlfile(fileName)`

Arguments

xml an xml character string to parse

fileName the name of the xml file to parse

Value

a Document object

Functions

- `parse.xmlstring`: create a Document from a character string
- `parse.xmlfile`: create a Document from a xml file

xmlr

*xmlr***Description**

A package for creating and reading and manipulating XML inspired by JDOM (<http://www.jdom.org/>), implemented with Reference Classes.

Examples

```
library("xmlr")
doc <- Document$new()
root <- Element$new("table")
root$setAttribute("xmlns", "http://www.w3.org/TR/html4/")
doc$setRootElement(root)

root$addContent(
  Element$new("tr")
  $addContent(Element$new("td")$setText("Apples"))
  $addContent(Element$new("td")$setText("Bananas"))
)
table <- doc$getRootElement()
stopifnot(table$getName() == "table")
stopifnot(table$getAttribute("xmlns") == "http://www.w3.org/TR/html4/")

children <- table$getChild("tr")$getChildren()
stopifnot(length(children) == 2)
stopifnot(children[[1]]$getText() == "Apples")
stopifnot(children[[2]]$getText() == "Bananas")

# you can also parse character strings (or parse a file using parse.xmlfile(fileName))
doc <- parse.xmlstring("<foo><bar><baz val='the baz attribute' /></bar></foo>")
```

xmlrToDataFrame

*Create a data frame from a xmlr Element***Description**

This is a convenience method to take all the children of the given Element and create a data frame based on the content of each child where each child constitutes a row and the attributes or elements (including text) will constitute the columns. It assumes a homogeneous structure and the column names are taken from the first child

Usage

```
xmlrToDataFrame(element)
```


Arguments

element the element to convert

Value

a data frame

Index

AbstractClass (AbstractClass-class), [2](#)
AbstractClass-class, [2](#)
as.character, Document-method
 (Document-class), [2](#)
as.character, Element-method
 (Element-class), [4](#)
as.character, Text-method (Text-class), [6](#)
as.vector, Document-method
 (Document-class), [2](#)
as.vector, Element-method
 (Element-class), [4](#)
as.vector, Text-method (Text-class), [6](#)

Content (Content-class), [2](#)
Content-class, [2](#)

Document (Document-class), [2](#)
Document-class, [2](#)
DomBuilder (DomBuilder-class), [3](#)
DomBuilder-class, [3](#)

Element (Element-class), [4](#)
Element-class, [4](#)

isRc, [5](#)

parse.xmlfile (xmlImporter), [7](#)
parse.xmlstring (xmlImporter), [7](#)
Parser (Parser-class), [6](#)
Parser-class, [6](#)

Stack (Stack-class), [6](#)
Stack-class, [6](#)

Text (Text-class), [6](#)
Text-class, [6](#)

xmlImporter, [7](#)
xmlr, [8](#)
xmlrToDataFrame, [8](#)