R4X: Convenient XML Manipulation for R

Romain Francois - Mango Solutions
rfrancois@mango-solutions.com

August 10, 2008

Abstract

The R4X package enhances the XML package by providing a simple mechanism to create, read and manipulate XML structures from within R. The functionality of the package is based on the E4X (EcmaScript for XML) add-on to the javascript specifications.

R4X implements templating of XML structures based on the brew package and convenient extraction of XML content using an XPath-like syntax.

The XML format is used to transfer data accross many applications, specially web applications. The R4X package came from the need of a convenient tool to manipulate XML data when R is part of a bigger application. The presentation will describe key features of the R4X package, including examples of using the functionality to build a simple RSS reader and a tag cloud generated by using description of all current CRAN packages.

Keywords: XML, templates, data manipulation

Creation of XML

The generic `xml` function allows creation of XML structures as defined by the `XML` package from character vectors, connections. Users can write other `xml` methods to convert different formats to XML structures

```
R> x <- xml( '
     <test>
        <foo blah="1"/>
        <bar/>
     </test>' )
R> x

<test>
 <foo blah="1"/>
 <bar/>
</test>

R> class( x)

[1] "XMLNode"
```

The character string used by the `xml` function may contain R code embedded in curly brackets.

```
R> y <- "something"
R> x <- xml( '
     <test>
        <foo blah="{y}"/>
        <bar/>
     </test>' )
R> x
```

```
<test>
 <foo blah="something"/>
 <bar/>
</test>
```

Extracting XML content

The `[` and `[[` methods from the `XML` package are masked by R4X to provide an XPath-like mechanism for extracting and modifying content of an XML structure.

```
R> x <- xml( '
   <root>
     <child id="1">
       <subchild id = "sub1" >foo</subchild>
       <subchild id = "sub2" >bar</subchild>
     </child>
     <child id="2">
       <subchild id="a">blah</subchild>
       <subchild id="b">bob</subchild>
       <something id="c" />
     </child>
     <fruits>
        <fruit>banana</fruit>
        <fruit>mango</fruit>
     </fruits>
   </root>
   ' )
R> # extracting a single XMLNode
R> x[ "fruits" ]
```

```
<fruits>
 <fruit>banana</fruit>
 <fruit>mango</fruit>
</fruits>
```

```
R> # extracting the text content of each <subchild>
R> x[ "child/subchild/#" ]
```

```
child.subchild child.subchild child.subchild child.subchild
         "foo"          "bar"         "blah"          "bob"
```

```
R> # extracting the id attribute of each <child>
R> x[ "child/@id" ]

child child
  "1"   "2"

R> # extracting the <subchild> of the second <child>
R> x[ "child[2]/subchild" ]

$subchild
<subchild id="a">blah</subchild>

$subchild
<subchild id="b">bob</subchild>
```

Example - Tag Cloud Generator

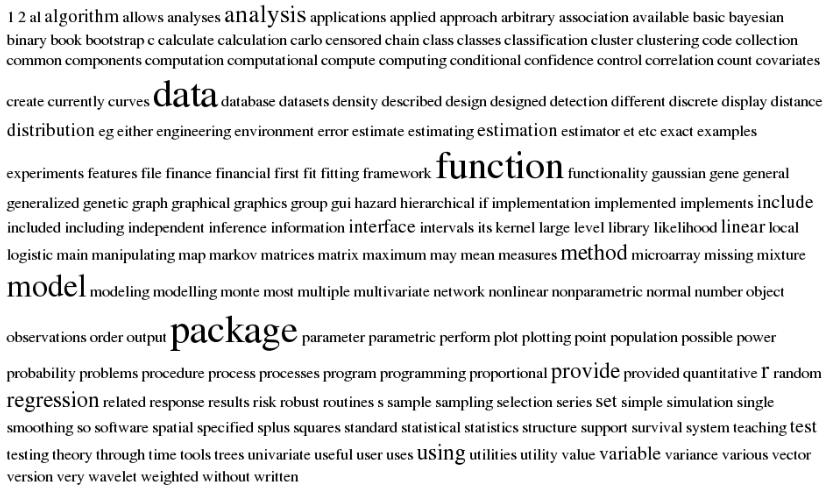
1 2 al algorithm allows analyses analysis applications applied approach arbitrary association available basic bayesian
binary book bootstrap c calculate calculation carlo censored chain class classes classification cluster clustering code collection
common components computation computational compute computing conditional confidence control correlation count covariates
create currently curves data database datasets density described design designed detection different discrete display distance
distribution eg either engineering environment error estimate estimating estimation estimator et etc exact examples
experiments features file finance financial first fit fitting framework function functionality gaussian gene general
generalized genetic graph graphical graphics group gui hazard hierarchical if implementation implemented implements include
included including independent inference information interface intervals its kernel large level library likelihood linear local
logistic main manipulating map markov matrices matrix maximum may mean measures method microarray missing mixture
model modeling modelling monte most multiple multivariate network nonlinear nonparametric normal number object
observations order output package parameter parametric perform plot plotting point population possible power
probability problems procedure process processes program programming proportional provide provided quantitative r random
regression related response results risk robust routines s sample sampling selection series set simple simulation single
smoothing so software spatial specified splus squares standard statistical statistics structure support survival system teaching test
testing theory through time tools trees univariate useful user uses using utilities utility value variable variance various vector
version very wavelet weighted without written

Figure 1: Tag cloud generated from words used in descriptions of CRAN packages using the R4X package.