

Package ‘R.rsp’

January 12, 2012

Version 0.7.1

Date 2011-11-28

Title Dynamic generation of scientific reports

Author Henrik Bengtsson <henrikb@braju.com>

Maintainer Henrik Bengtsson <henrikb@braju.com>

Depends R (>= 2.5.0), R.oo (>= 1.8.3), R.utils (>= 1.9.6)

Suggests tcltk

Description An RSP document is a text-based document containing an R-embedded template of the final document, e.g. “Today’s date is `<%=Sys.Date()%>`”. An RSP document is preprocessed, parsed and translated into an R script, which when sourced generates the final document. This way it is possible to dynamically generate reports in plain text, HTML, TeX etc, e.g. `“\{\}includegraphics{<%=toPDF('Normal', { curve(dnorm, from=5,to=+5) })%>}`”. It can also be used to enhance other literate programming languages such as Sweave, e.g. `“<<eval=<%=doEval%>>>= [...] @”`. As explained in one of the vignettes, RSP-embedded LaTeX vignettes can easily be included in any R package. In addition to RSP, this package also provides an internal cross-platform web server and built-in dynamic RSP-embedded HTML help pages, which can be launched by `browseRsp()`. If other packages provide RSP help pages, these are automatically linked to as well.

License LGPL (>= 2.1)

URL <http://www.braju.com/R/>

LazyLoad TRUE

Repository CRAN

Date/Publication 2012-01-12 09:24:13

R topics documented:

R.rsp-package	2
browseRsp	3
FileRspResponse	4
HtmlRspLanguage	5
HttpDaemon	6
HttpDaemonRspResponse	8
HttpRequest	10
rsp	11
RspLanguage	13
RspPage	14
RspResponse	15

Index	17
--------------	-----------

R.rsp-package	<i>Package R.rsp</i>
---------------	----------------------

Description

An RSP document is a text-based document containing an R-embedded template of the final document, e.g. "Today's date is <

Requirements

This is a cross-platform package implemented in plain R. This package depends on the packages **R.oo** [1] and **R.utils**.

Note that no webserver is required to process RSP documents.

Installation

To install this package, do `install.packages("R.rsp")`.

To get started

To get started, see:

1. `rsp()` - To compile any RSP-embedded document.
2. `browseRsp()` - Launches a locally running RSP website powered by an internal web server and RSP-embedded HTML pages. From this page you access not only help pages and demos on how to use RSP, but also other package RSP pages.

Wishlist

Here is a list of features that would be useful, but which I have too little time to add myself. Contributions are appreciated.

- Extract the HTTP daemon part of this package and create a standalone package named R.httpd or similar. It should provide a method to register simple modules, such as an RSP module. The R.rsp package should then only be a simple module.
- Write "plugins" to common web servers, e.g. modules to the Apache webserver.
- Add support for multiple default files; needs Tcl coding.
- Create a root ServletRequest class to support not only HTTP requests, but also other types of request, e.g. FileRequest etc. This requires some thinking of user cases and design.

If you consider implement some of the above, make sure it is not already implemented by downloading the latest "devel" version!

License

The releases of this package is licensed under LGPL version 2.1 or newer.

The development code of the packages is under a private licence (where applicable) and patches sent to the author fall under the latter license, but will be, if incorporated, released under the "release" license above.

How to cite this package

Not available.

References

- 1 H. Bengtsson, *The R.oo package - Object-Oriented Programming with References Using Standard R Code*, In Kurt Hornik, Friedrich Leisch and Achim Zeileis, editors, Proceedings of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003), March 20-22, Vienna, Austria. <http://www.ci.tuwien.ac.at/Conferences/DSC-2003/Proceedings/>

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

browseRsp

Starts the internal web browser and opens the URL in the default web browser

Description

Starts the internal web browser and opens the URL in the default web browser. From this page you access not only help pages and demos on how to use RSP, but also other package RSP pages.

Usage

```
## Default S3 method:
browseRsp(url=sprintf("http://%s:%d/%s", host, port, path), host="127.0.0.1", port=8074, path="", star
```

Arguments

url	A character string for the URL to be viewed. By default the URL is constructed from the host, port, and the path parameters.
host	An optional character string for the host of the URL.
port	An optional integer for the port of the URL.
path	An optional character string for the context path of the URL.
start	If TRUE , the internal R web server is started if not already started, otherwise not.
stop	If TRUE , the internal R web server is stopped, if started.
...	Additional arguments passed to browseURL .

Value

Returns nothing.

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

See Also

Internally, [browseURL](#) is used to launch the browser.

FileRspResponse	<i>The FileRspResponse class</i>
-----------------	----------------------------------

Description

Package: R.rsp

Class FileRspResponse**Object**

```
~~|
~~+--RspResponse
~~~~~|
~~~~~+--FileRspResponse
```

Directly known subclasses:

[HttpDaemonRspResponse](#)

```
public static class FileRspResponse
extends RspResponse
```

Usage

```
FileRspResponse(file=stdout(), path=NULL, overwrite=FALSE, ...)
```

Arguments

file	A filename or a connection to write responses to.
path	An optional path to the file.
overwrite	If <code>FALSE</code> , an error is thrown if the output file already exists, otherwise not.
...	Not used.

Fields and Methods**Methods:**

flush	Flushes the response buffer.
getAbsolutePath	Gets the absolute pathname to the current RSP file.
getName	Gets the (base)name of the current RSP file.
getOutput	Gets the output for an RSP response.
getPath	Gets the path of the directory of the current RSP file.
write	Writes an RSP response to the predefined output file.

Methods inherited from RspResponse:

flush, import, write

Methods inherited from Object:

\$, \$<-, [], [[<-, as.character, attach, attachLocally, clearCache, clearLookupCache, clone, detach, equals, extend, finalize, gc, getEnvironment, getFieldModifier, getFieldModifiers, getFields, getInstantiationTime, getStaticInstance, hasField, hashCode, ll, load, objectSize, print, registerFinalizer, save

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

HtmlRspLanguage

The HtmlRspLanguage class

Description

Package: R.rsp

Class HtmlRspLanguage

[Object](#)

~~|

~~+--RspLanguage

```
~~~~~|
~~~~~+---HtmlRspLanguage
```

Directly known subclasses:

```
public static class HtmlRspLanguage
extends RspLanguage
```

Usage

```
HtmlRspLanguage(...)
```

Arguments

```
... Arguments passed to the constructor of the RspLanguage.
```

Fields and Methods

Methods:

escape	Escapes a string specifically for the HTML language.
getComment	Gets a comment string specifically for the HTML language.
getVerbatim	Gets a verbatim string specifically for the HTML language.

Methods inherited from [RspLanguage](#):

[escape](#), [getComment](#), [getLanguage](#), [getNewline](#), [getVerbatim](#)

Methods inherited from [Object](#):

[\\$](#), [\\$<-](#), [\[\[](#), [\[\[<-](#), [as.character](#), [attach](#), [attachLocally](#), [clearCache](#), [clearLookupCache](#), [clone](#), [detach](#), [equals](#), [extend](#), [finalize](#), [gc](#), [getEnvironment](#), [getFieldModifier](#), [getFieldModifiers](#), [getFields](#), [getInstantiationTime](#), [getStaticInstance](#), [hasField](#), [hashCode](#), [ll](#), [load](#), [objectSize](#), [print](#), [registerFinalizer](#), [save](#)

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

HttpDaemon

The HttpDaemon class

Description

Package: R.rsp

Class HttpDaemon

[Object](#)

~~|

~~+--HttpDaemon

Directly known subclasses:

public static class **HttpDaemon**

extends [Object](#)

A minimalistic HTTP daemon (web server) that also preprocesses RSP.

Usage

```
HttpDaemon(...)
```

Arguments

... Not used.

Details

The actual server is written in Tcl such that it runs in a non-blocking mode, which means that the R prompt will be available for other things. This class is tightly coupled with the source code of the Tcl script.

For security reasons, the server only accept connections from the local host (127.0.0.1). This lowers the risk for external computers to gain access to the R session. This is asserted by the `accept_connect` Tcl procedure in `r-httpd.tcl` (located in `system("tcl/", package="R.rsp")`). If access from other hosts are wanted, then this procedure needs to be modified.

The Tcl server was written by Steve Uhlers, and later adopted for R by Philippe Grosjean and Tom Short (Rpad package author) [1].

Fields and Methods

Methods:

appendRootPaths	Appends and inserts new paths to the list of known root directories.
as.character	Returns a short string describing the HTTP daemon.
finalize	-
getConfig	Retrieves the server's 'config' structure from Tcl.
getCount	-
getDefaultFilename	Gets the default filename to be loaded by the HTTP daemon.
getHttpRequest	Gets the HTTP request.
getPort	Gets the socket port of the HTTP daemon.
getRootPaths	Gets the root directories of the HTTP daemon.
insertRootPaths	-
isStarted	Checks if the HTTP daemon is started.

<code>openUrl</code>	Starts the HTTP daemon and launches the specified URL.
<code>processRsp</code>	Processes an RSP page.
<code>restart</code>	Restarts the HTTP daemon.
<code>setCount</code>	-
<code>setRootPaths</code>	Sets a new set of root directories for the HTTP daemon.
<code>sourceTcl</code>	Loads the Tcl source for the HTTP daemon into R.
<code>start</code>	Starts the HTTP daemon.
<code>startHelp</code>	Starts the HTTP daemon and launches the help page.
<code>stop</code>	Stops the HTTP daemon.
<code>writeResponse</code>	Writes a string to the HTTP output connection.

Methods inherited from Object:

\$, \$<-, [[, [[<-, as.character, attach, attachLocally, clearCache, clearLookupCache, clone, detach, equals, extend, finalize, gc, getEnvironment, getFieldModifier, getFieldModifiers, getFields, getInstantiationTime, getStaticInstance, hasField, hashCode, ll, load, objectSize, print, registerFinalizer, save

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

References

[1] Rpad package, Tom Short, 2005.

Examples

```
if (interactive()) {
  if (!HttpDaemon$isStarted()) {
    # Start the web server
    rootPath <- system.file("rsp", package="R.rsp")
    HttpDaemon$start(rootPath=rootPath, port=8074, default="index.rsp")
  }

  browseURL("http://127.0.0.1:8074/")
}
```

HttpDaemonRspResponse *The HttpDaemonRspResponse class*

Description

Package: R.rsp
Class HttpDaemonRspResponse

Object

~~|

```

~+---RspResponse
~~~~~|
~~~~~+---FileRspResponse
~~~~~|
~~~~~+---HttpDaemonRspResponse

```

Directly known subclasses:

```

public static class HttpDaemonRspResponse
extends FileRspResponse

```

An instance of class `HttpDaemonRspResponse`, which extends the `RspResponse` class, is a buffer for output (response) sent to an `HttpDaemon`. It provides a method `write()` for writing output and a method `flush()` for flush the written output to the HTTP daemon.

Usage

```
HttpDaemonRspResponse(httpDaemon=NULL, ...)
```

Arguments

<code>httpDaemon</code>	An <code>HttpDaemon</code> object.
<code>...</code>	Not used.

Details

The purpose of this method is to provide partial writing of HTTP response such that, for instance, a web browser can display parts of an HTML page while the rest is generated. Note that this is only supported by the HTTP v1.1 protocol.

Note: The minimalistic HTTP daemon (written in Tcl) used internally currently only supports HTTP v1.0. In other words, although this class is used already, the output is only flushed at the end.

Fields and Methods

Methods:

<code>flush</code>	Flushes the buffer of an <code>HttpDaemonRspResponse</code> to the <code>HttpDaemon</code> .
<code>write</code>	Writes strings to an <code>HttpDaemonRspResponse</code> buffer.

Methods inherited from `FileRspResponse`:

`flush`, `getAbsolutePath`, `getName`, `getOutput`, `getPath`, `write`

Methods inherited from `RspResponse`:

`flush`, `import`, `write`

Methods inherited from `Object`:

\$, \$<-, [], [[<-, as.character, attach, attachLocally, clearCache, clearLookupCache, clone, detach, equals, extend, finalize, gc, getEnvironment, getFieldModifier, getFieldModifiers, getFields, getInstantiationTime, getStaticInstance, hasField, hashCode, ll, load, objectSize, print, registerFinalizer, save

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

See Also

[HttpDaemon](#).

HttpRequest

The HttpRequest class

Description

Package: R.rsp

Class HttpRequest

[Object](#)

~~|

~~+--HttpRequest

Directly known subclasses:

```
public static class HttpRequest
```

```
extends Object
```

Usage

```
HttpRequest(requestUri=NULL, parameters=list(), ...)
```

Arguments

requestUri A [character](#) string of the requested URI.

parameters A named [list](#) of parameter values.

... Not used.

Fields and Methods

Methods:

<code>as.character</code>	Returns a short string describing the HTTP request.
<code>getContentLength</code>	Gets the length of contents.
<code>getContentType</code>	Gets the MIME type of the body of the request.
<code>getContextPath</code>	-
<code>getDateHeader</code>	-
<code>getHeader</code>	-
<code>getParameter</code>	Gets a parameter.
<code>getParameters</code>	Gets all parameters.
<code>getProtocol</code>	Gets the name and version of the protocol used to make this request.
<code>getQueryString</code>	-
<code>getRealPath</code>	Gets the file system path for a given URI.
<code>getRemoteAddress</code>	Gets the IP address of the client that sent the request.
<code>getRemoteHost</code>	Gets the fully qualified name of the client that sent the request.
<code>getRemoteUser</code>	-
<code>getRequestUri</code>	-
<code>getRequestUrl</code>	-
<code>getScheme</code>	Gets the scheme used to make this request.
<code>getServerName</code>	Gets the host name of the server that reviewed the request.
<code>getServerPort</code>	Gets the port number on which this request was received.
<code>getServletPath</code>	-
<code>hasParameter</code>	Checks if a parameter exists.
<code>nbrOfParameters</code>	Gets the number of parameters.

Methods inherited from Object:

`$`, `$<-`, `[[`, `[[<-`, `as.character`, `attach`, `attachLocally`, `clearCache`, `clearLookupCache`, `clone`, `detach`, `equals`, `extend`, `finalize`, `gc`, `getEnvironment`, `getFieldModifier`, `getFieldModifiers`, `getFields`, `getInstantiationTime`, `getStaticInstance`, `hasField`, `hashCode`, `ll`, `load`, `objectSize`, `print`, `registerFinalizer`, `save`

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

rsp

Compiles an RSP document

Description

Compiles an RSP document.

Usage

Default S3 method:

```
rsp(filename=NULL, path=NULL, text=NULL, response=NULL, ..., envir=parent.frame(), postprocess=TRUE, v
```

Arguments

filename, path	The filename and (optional) path of the RSP document to be compiled.
text	A character vector of RSP code to be processed, iff argument filename is not given.
response	Specifies where the final output should be sent. If argument text is given, then stdout() is used. Otherwise, the output defaults to that of the type-specific compiler.
...	Additional arguments passed to the type-specific compiler.
envir	The environment in which the RSP document is evaluated.
postprocess	If TRUE , and a postprocessing method exists for the generated document type, it is postprocessed as well.
verbose	See Verbose .

Value

Returns what the type-specific compiler returns.

Postprocessing

For some document types, the `rsp()` method automatically postprocesses the generated document as well.

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

Examples

```
# EXAMPLE #1
rsp(text="A random number in [1,100]: <%=sample(1:100, size=1)%>\n")

# EXAMPLE #2
rsp(text="A random number in [1,100] (assigned to 'x' in the workspace): <%= x <- sample(1:100, size=1); %>\n")
printf("The random number generated by the RSP document was: %d\n", x)

# EXAMPLE #3
path <- system.file("doc", package="R.rsp")
pathname <- rsp("Dynamic_LaTeX_reports_with_RSP.tex.rsp", path=path)
printf("Created document: %s\n", pathname)
```

RspLanguage

The RspLanguage class

Description

Package: R.rsp

Class RspLanguage

[Object](#)

~~|

~~+--RspLanguage

Directly known subclasses:

[HtmlRspLanguage](#)

```
public static class RspLanguage
```

```
extends Object
```

An RspLanguage object specifies what the markup language of the response/output document is, e.g. plain text and HTML. The RspLanguage class provides methods to obtain language specific strings/output such as how newlines and comments are written. The RspLanguage class describes a plain text languages. For HTML see the [HtmlRspLanguage](#) subclass.

Usage

```
RspLanguage(language="plain", ...)
```

Arguments

language A [character](#) string.

... Not used.

Fields and Methods

Methods:

escape	Escapes a string specifically for a given RSP response language.
getComment	Gets a comment string specific for a given RSP response language.
getLanguage	Gets the language string.
getNewline	Gets the newline string specific for a given RSP response language.
getVerbatim	Gets a verbatim string specific for a given RSP response language.

Methods inherited from [Object](#):

\$, \$<-, [], [[<-, as.character, attach, attachLocally, clearCache, clearLookupCache, clone, detach, equals, extend, finalize, gc, getEnvironment, getFieldModifier, getFieldModifiers, getFields, getInstantiationTime, getStaticInstance, hasField, hashCode, ll, load, objectSize, print, registerFinalizer, save

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

RspPage

The RspPage class

Description

Package: R.rsp

Class RspPage

[Object](#)

~~|

~~+--RspPage

Directly known subclasses:

public static class **RspPage**
 extends [Object](#)

Usage

```
RspPage(pathname=NULL, ...)
```

Arguments

pathname	A character string.
...	Not used.

Fields and Methods

Methods:

getAbsolutePath	Gets the absolute pathname to the current RSP file.
getName	Gets the (base)name of the current RSP file.
getPath	Gets the path of the directory of the current RSP file.

Methods inherited from Object:

\$, \$<-, [], []<-, as.character, attach, attachLocally, clearCache, clearLookupCache, clone, detach, equals, extend, finalize, gc, getEnvironment, getFieldModifier, getFieldModifiers, getFields, getInstantiationTime, getStaticInstance, hasField, hashCode, ll, load, objectSize, print, registerFinalizer, save

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

RspResponse

The RspResponse class

Description

Package: R.rsp

Class RspResponse**Object**

~~|

~~+--RspResponse

Directly known subclasses:

[FileRspResponse](#), [HttpDaemonRspResponse](#)

public abstract static class **RspResponse**

extends [Object](#)

An abstract class that provides basic methods to write and flush output to the generated document.

Usage

RspResponse(...)

Arguments

... Not used.

Fields and Methods**Methods:**

[flush](#) Flushes the response buffer.
[import](#) Imports the output from another RSP file.
[write](#) Writes an RSP response to the predefined output file.

Methods inherited from Object:

\$, \$<-, [], []<-, as.character, attach, attachLocally, clearCache, clearLookupCache, clone, detach, equals, extend, finalize, gc, getEnvironment, getFieldModifier, getFieldModifiers, getFields, getInstantiationTime, getStaticInstance, hasField, hashCode, ll, load, objectSize, print, registerFinalizer, save

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

Index

*Topic **IO**

- browseRsp, 3
- HttpDaemon, 6
- HttpDaemonRspResponse, 8
- rsp, 11

*Topic **classes**

- FileRspResponse, 4
- HtmlRspLanguage, 5
- HttpDaemon, 6
- HttpDaemonRspResponse, 8
- HttpRequest, 10
- RspLanguage, 13
- RspPage, 14
- RspResponse, 15

*Topic **file**

- browseRsp, 3
- rsp, 11

*Topic **package**

- R.rsp-package, 2

- appendRootPaths, 7
- as.character, 7, 11

- browseRsp, 2, 3
- browseURL, 4

- character, 4, 10, 12–14
- connection, 5

- environment, 12
- escape, 6, 13

- FALSE, 5
- FileRspResponse, 4, 9, 15
- flush, 5, 9, 15

- getAbsolutePath, 5, 14
- getComment, 6, 13
- getConfig, 7
- getContentLength, 11
- getContentType, 11

- getDefaultFilename, 7
- getHttpRequest, 7
- getLanguage, 13
- getName, 5, 14
- getNewline, 13
- getOutput, 5
- getParameter, 11
- getParameters, 11
- getPath, 5, 14
- getPort, 7
- getProtocol, 11
- getRealPath, 11
- getRemoteAddress, 11
- getRemoteHost, 11
- getRootPaths, 7
- getScheme, 11
- getServerName, 11
- getServerPort, 11
- getVerbatim, 6, 13
- hasParameter, 11
- HtmlRspLanguage, 5, 13
- HttpDaemon, 6, 9, 10
- HttpDaemonRspResponse, 4, 8, 15
- HttpRequest, 10

- import, 15
- integer, 4
- isStarted, 7

- list, 10

- nbrOfParameters, 11

- Object, 4, 5, 7, 8, 10, 13–15
- openUrl, 8

- processRsp, 8

- R.rsp (R.rsp-package), 2
- R.rsp-package, 2

restart, 8
rsp, 2, 11
RspLanguage, 5, 6, 13
RspPage, 14
RspResponse, 4, 9, 15

setRootPaths, 8
sourceTcl, 8
start, 8
startHelp, 8
stdout, 12
stop, 8

TRUE, 4, 12

vector, 12
Verbose, 12

write, 5, 9, 15
writeResponse, 8