Package ‘BrowserViz’

March 28, 2017

Type Package

Title BrowserViz: interactive R/browser graphics using websockets and JSON

Version 1.6.0

Date 2016-10-12

Author Paul Shannon

Maintainer Paul Shannon <pshannon@systemsbiology.org>

Depends R (>= 3.2.1), jsonlite (>= 0.9.15), httpuv(>= 1.3.2)

Imports methods, BiocGenerics

Suggests RUnit, BiocStyle

Description Interactive graphics in a web browser from R, using websockets and JSON.

License GPL-2

LazyLoad yes

biocViews Visualization, ThirdPartyClient

NeedsCompilation no

R topics documented:

BrowserVizClass

Index

BrowserVizClass

BrowserViz: a base class (for extension) and standalone example of R/JavaScript interactive web browser visualization

Description

A concrete base class for interactive R/javascript visualization tools. Derived classes obtain socket setup, status and retrieval methods for free, obscuring many complicated details.
Usage

BrowserViz(portRange, host="localhost", title="BrowserViz", quiet=TRUE,
browserFile=NA, httpQueryProcessingFunction=NULL)
toJSON(..., auto_unbox=TRUE)
addRMessageHandler(key, function)

## S4 method for signature 'BrowserVizClass'
ready(obj)
## S4 method for signature 'BrowserVizClass'
send(obj, msg)
## S4 method for signature 'BrowserVizClass'
browserResponseReady(obj)
## S4 method for signature 'BrowserVizClass'
getBrowserResponse(obj)
## S4 method for signature 'BrowserVizClass'
getBrowserInfo(obj)
## S4 method for signature 'BrowserVizClass'
closeWebSocket(obj)
## S4 method for signature 'BrowserVizClass'
port(obj)
## S4 method for signature 'BrowserVizClass'
getBrowserWindowTitle(obj)
## S4 method for signature 'BrowserVizClass'
setBrowserWindowTitle(obj, newTitle, proclaim=FALSE)
## S4 method for signature 'BrowserVizClass'
getBrowserWindowSize(obj)

Arguments

obj The BrowserVizClass object returned by the class constructor.
portRange One or more consecutive integers in the range 1025-65535. A typical choice
is 9000:9024. The BrowserViz class constructor will try these one at a time in
succession until a free port is found and the connection to your web browser is
established. If no open ports are found in the supplied range, an error is reported.
host Nearly always left to its default value, "localhost" but included as a parameter
supporting remote computers for future flexibility.
title The constructor creates a new window (or a new tab, depending on how you web
browser is configured). This title is displayed at the top of the window or tab.
quiet Trace and tracking messages are written to the R console if this variable is set to
FALSE.
browserFile defaults to NA, which is interpreted as an instruction to use viz.html in the
inst directory of this package. Every subclass will have its own (possibly quite
complex) browserFile, containing HTML, Javascript, CSS, and calls to powerful
Javascript libraries (i.e., jQuery, d3, cytoscape.js). viz.html provides Javascript
endpoints of, for instance, the getBrowserWindowTitle and setBrowserWindowTitle
class methods described above.
httpQueryProcessingFunction
defaults to NULL. When not NULL this function, supplied by the subclass, is
called whenever an HTTP (as opposed to a websocket) request arrives. Thus
the BrowserViz subclass can transfer data to the web browser using a traditional
HTTP GET when that is advantageous.

msg  A name list, with four required slots: "cmd", "status", "callback", "payload".
See below.

newTitle  A character string.

proclaim  Logical, default FALSE; if TRUE will add newTitle to the web page's body,
providing vivid evidence of R controlling the browser.

key  A character string, the "cmd" field of the incoming four-field JSON command,
used to dispatch on, so that the proper function is called.

functionName  A character string: the name of a function to which incoming web socket json
commands can be dispatched.

auto_unbox  Logical, default TRUE; unboxing: do not coerce a scalar into a 1-element list,
as the new (2015) jsonlite package prefers to do.

Arguments for our local, very slightly redefined version of toJSON.

Methods

In the code snippets below, obj is an instance of the BrowserVizClass.

BrowserViz(portRange, host="localhost", title="BrowserViz", quiet=TRUE, browserFile=NA, httpQueryProcessingFunction=NULL)
Constructs a BrowserViz object. Among the several actions included are: your default web-
browser browses to the uri of a minimal http server embedded in BrowserViz; the
browserFile is returned to the browser; the websocket connection is initialized on both ends, and the lowest
numbered port in portRange.

ready(obj): returns TRUE when the R/browser websocket connection is ready for use.

port(obj): returns the actual port being used.

getBrowserInfo(obj): returns a character string describing the browser to which we are
connected, using the standard W3C DOM navigator.userAgent.

send(obj, msg): sends a properly structured (having four fields: cmd, callback, status, payload)
JSON message to the browser.

browserResponseReady(obj): returns TRUE when the asynchronous response to the last mes-
sage has been received from the browser.

getBrowserResponse(obj): returns the just-received JSON-encoded, four-field response to the
latest message sent to the browser.

closeWebSocket(obj): Close the websocket port now in use, making it available for reuse.

getWindowTitle(obj): Returns the title of the web page (or tab).

setWindowTitle(obj, newTitle, proclaim=FALSE): Sets the title of the web page
or tab to which we are currently connected. The "proclaim" argument is for demonstration
purposes only, illustrating to new users that the web page can be interactively manipulated
from R.

getTitle(obj): in pixels.

Further arguments for toJSON, typically just the variable to be encoded.

Author(s)

Paul Shannon
Examples

library(BrowserViz)

bv <- BrowserViz(4000:4024)

## make sure everything is ready to use
while(!ready(bv)) Sys.sleep(0.1)

port(bv)

## illustrate a "low level" call. This detail is usually hidden from
## the user, implemented and contained (in the case of this example)
## in a getBrowserWindowTitle(bv) method call. This level of detail
## reveals what goes on behind the scenes.

msg <- list(cmd="getWindowTitle", status="request", callback="handleResponse", payload="")
send(bv, msg)
while(!browserResponseReady(bv)) Sys.sleep(0.1)
getBrowserResponse(bv)

## a simpler user-level approach:
getBrowserWindowTitle(bv)

## set and get the windowTitle
setBrowserWindowTitle(bv, "new title", proclaim=TRUE)
getBrowserWindowTitle(bv)

## BrowserViz provides another information method which, like the others, will apply
## and may be of some use to derived classes
getBrowserWindowSize(bv)

## finally, you should close BrowserViz when you are done, returning
## the port for use by other applications.
closeWebSocket(bv)
Index

*Topic **classes**
  BrowserVizClass, 1
*Topic **methods**
  BrowserVizClass, 1

addRMessageHandler (BrowserVizClass), 1
browserResponseReady (BrowserVizClass), 1
browserResponseReady, BrowserVizClass-method (BrowserVizClass), 1
BrowserViz (BrowserVizClass), 1
BrowserVizClass, 1
BrowserVizClass-class (BrowserVizClass), 1
class:BrowserVizClass (BrowserVizClass), 1
closeWebSocket (BrowserVizClass), 1
closeWebSocket, BrowserVizClass-method (BrowserVizClass), 1

getBrowserInfo (BrowserVizClass), 1
getBrowserInfo, BrowserVizClass-method (BrowserVizClass), 1
getBrowserResponse (BrowserVizClass), 1
getBrowserResponse, BrowserVizClass-method (BrowserVizClass), 1
getBrowserWindowSize (BrowserVizClass), 1
getBrowserWindowSize, BrowserVizClass-method (BrowserVizClass), 1
getBrowserWindowTitle (BrowserVizClass), 1
getBrowserWindowTitle, BrowserVizClass-method (BrowserVizClass), 1

port (BrowserVizClass), 1
port, BrowserVizClass-method (BrowserVizClass), 1

ready (BrowserVizClass), 1
ready, BrowserVizClass-method (BrowserVizClass), 1

send (BrowserVizClass), 1
send, BrowserVizClass-method (BrowserVizClass), 1
setBrowserWindowTitle (BrowserVizClass), 1
setBrowserWindowTitle, BrowserVizClass-method (BrowserVizClass), 1
show, BrowserVizClass-method (BrowserVizClass), 1
toJSON (BrowserVizClass), 1