

Package ‘BrowserVizDemo’

April 14, 2017

Type Package

Title BrowserVizDemo: How to subclass BrowserViz

Version 1.6.0

Date 2016-06-07

Author Paul Shannon

Maintainer Paul Shannon <pshannon@systemsbiology.org>

Depends R (>= 3.2.3), BrowserViz, Rcpp (>= 0.11.5), jsonlite (>= 0.9.15), httpuv(>= 1.3.2)

Imports methods, BiocGenerics

Suggests RUnit, BiocStyle

Description A BrowserViz subclassing example, xy plotting in the browser using d3.

License GPL-2

LazyLoad yes

biocViews Visualization, ThirdPartyClient

NeedsCompilation no

R topics documented:

BrowserVizDemoClass 1

Index 4

BrowserVizDemoClass *BrowserVizDemo: Interactive R/browser plotting*

Description

An early, simple example of how to create useful interactive graphics in a class derived from BrowserViz. This package could evolve to be a drop-in replacement for the R base "plot" function, for plotting xy values. It has the additional virtue of full interactivity on the plotting surface, which is here an HTML5/d3 canvas. Manually selected points on that canvas, for example, can be queried in R. This may facilitate exploratory data analysis.

Usage

```
BrowserVizDemo(portRange, host="localhost", title="BrowserVizDemo", quiet=TRUE)

## S4 method for signature 'BrowserVizDemoClass'
plot(obj, x, y)
## S4 method for signature 'BrowserVizDemoClass'
getSelection(obj)
```

Arguments

<code>obj</code>	The BrowserVizDemoClass object returned by the class constructor.
<code>x</code>	A numeric vector, the x-coordinates of the points to plot.
<code>y</code>	A numeric vector, the y-coordinates of the points to plot.
<code>portRange</code>	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.
<code>host</code>	Nearly always left to its default value, "localhost" but included as a parameter supporting remote computers for future flexibility.
<code>title</code>	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.
<code>quiet</code>	Trace and tracking messages are written to the R console if this variable is set to FALSE.

Methods

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

```
BrowserVizDemo(portRange, host="localhost", title="BrowserVizDemo", quiet=TRUE, browserFile=NA)
```

Constructs a BrowserVizDemo object. Among the several actions included are: your default webbrowser browses to the uri of a minimal http server embedded in BrowserVizDemo; the browserFile is returned to the browser; the websocket connection is initialized on both ends, and the lowest numbered port in portRange.

`plot(obj, x, y)`: Draws an interactive xy plot in your browser window, with labeled axes, and the surface scaled to the x and y coordinates. In time this method will mimic the rich behavior of the base R plot method, and all of its optional parameters.

Author(s)

Paul Shannon

Examples

```
library(BrowserVizDemo)

plotter <- BrowserVizDemo(4000:4024)

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

## plot a simple set of x-y paris
```

```
plot(plotter, 1:10, (1:10)^2)

## learn which port we are using
port(plotter)

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

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

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

## set and get the windowTitle
setBrowserWindowTitle(plotter, "new title")
getBrowserWindowTitle(plotter)

## BrowserVizDemo provides another information method which, like the others, will apply
## and maybe be of some use to derived classes
getBrowserWindowSize(plotter)

## finally, you should close BrowserVizDemo when you are done, returning
## the port for use by other applications.
closeWebSocket(plotter)
```

Index

*Topic **classes**

BrowserVizDemoClass, [1](#)

*Topic **methods**

BrowserVizDemoClass, [1](#)

BrowserVizDemo (BrowserVizDemoClass), [1](#)

BrowserVizDemoClass, [1](#)

BrowserVizDemoClass-class
(BrowserVizDemoClass), [1](#)

class:BrowserVizDemoClass
(BrowserVizDemoClass), [1](#)

getSelection (BrowserVizDemoClass), [1](#)

getSelection, BrowserVizDemoClass-method
(BrowserVizDemoClass), [1](#)

plot (BrowserVizDemoClass), [1](#)

plot, BrowserVizDemoClass-method
(BrowserVizDemoClass), [1](#)

show, BrowserVizDemoClass-method
(BrowserVizDemoClass), [1](#)