Package 'leafpop'

December 9, 2019

Type Package

Title Include Tables, Images and Graphs in Leaflet Pop-Ups

Version 0.0.5

Date 2019-12-09

Maintainer Tim Appelhans <tim.appelhans@gmail.com>

Description Creates 'HTML' strings to embed tables, images or graphs in pop-ups
of interactive maps created with packages like 'leaflet' or 'mapview'. Handles
local images located on the file system or via remote URL. Handles graphs created
with 'lattice' or 'ggplot2' as well as interactive plots created with 'htmlwidgets'.

License MIT + file LICENSE

Encoding UTF-8

Imports base64enc, brew, htmltools, htmlwidgets, sf, svglite, uuid

Suggests lattice, leaflet (>= 2.0.0), sp

RoxygenNote 7.0.2

NeedsCompilation no

Author Tim Appelhans [cre, aut],
Florian Detsch [aut]

Repository CRAN

Date/Publication 2019-12-09 19:50:02 UTC

R topics documented:

---

leafpop-package   *Include Tables, Images and Graphs in Leaflet Popups*

Description

Include Tables, Images and Graphs in Leaflet Popups

Details

Creates HTML strings to embed tables, images or graphs in popups of interactive maps created
with packages 'leaflet' or 'mapview'. Handles local paths to images on the file system or remote
urls. Handles graphs created with 'base' graphics, 'lattice' or 'ggplot2' as well as interactive plots
created with 'htmlwidgets'.

Author(s)

Tim Appelhans, Florian Detsch *Maintainer:* Tim Appelhans `<tim.appelhans@gmail.com>`

---

addPopupGraphs   *Add graph/plot popups to leaflet layers.*

Description

Add graph/plot popups to leaflet layers.

Usage

```
addPopupGraphs(map, graph, group, width = 300, height = 300)
```

Arguments

map	the `leaflet` map to add the popups to.
graph	A `list` of lattice or ggplot2 objects. Needs to be a list, even for a single plot!
group	the map group to which the popups should be added.
width	the width of the graph(s) in pixels.
height	the height of the graph(s) in pixels.

Value

A `leaflet` map.

Examples

```
if (interactive()) {
library(sf)
library(leaflet)

pt = data.frame(x = 174.764474, y = -36.877245)
pt = st_as_sf(pt, coords = c("x", "y"), crs = 4326)

p2 = levelplot(t(volcano), col.regions = terrain.colors(100))

leaflet() %>%
  addTiles() %>%
  addCircleMarkers(data = pt, group = "pt") %>%
  addPopupGraphs(list(p2), group = "pt", width = 300, height = 400)

}
```

---

addPopupImages   *Add image popups to leaflet layers.*

Description

Add image popups to leaflet layers.

Usage

```
addPopupImages(map, image, group, width = NULL, height = NULL)
```

Arguments

map	the leaflet map to add the popups to.
image	A character vector of file path(s) or web-URL(s) to any sort of image file(s).
group	the map group to which the popups should be added.
width	the width of the image(s) in pixels.
height	the height of the image(s) in pixels.

Value

A leaflet map.

Examples

```
if (interactive()) {
## remote images -----
### one image
library(leaflet)
library(sf)

pnt = st_as_sf(data.frame(x = 174.764474, y = -36.877245),
               coords = c("x", "y"),
               crs = 4326)

img = "http://bit.ly/1TVwRiR"

leaflet() %>%
  addTiles() %>%
  addCircleMarkers(data = pnt, group = "pnt") %>%
  addPopupImages(img, group = "pnt")

### multiple file (types)
library(sf)
images = c(img,
          "https://upload.wikimedia.org/wikipedia/commons/9/91/Octicons-mark-github.svg",
             "https://www.r-project.org/logo/Rlogo.png",
             "https://upload.wikimedia.org/wikipedia/commons/d/d6/MeanMonthlyP.gif")

pt4 = data.frame(x = jitter(rep(174.764474, 4), factor = 0.01),
                 y = jitter(rep(-36.877245, 4), factor = 0.01))
pt4 = st_as_sf(pt4, coords = c("x", "y"), crs = 4326)

leaflet() %>%
  addTiles() %>%
  addMarkers(data = pt4, group = "points") %>%
  addPopupImages(images, group = "points", width = 400) # NOTE the gif animation

## local images -----
pnt = st_as_sf(data.frame(x = 174.764474, y = -36.877245),
               coords = c("x", "y"), crs = 4326)
img = system.file("img","Rlogo.png",package="png")
leaflet() %>%
  addTiles() %>%
  addCircleMarkers(data = pnt, group = "pnt") %>%
  addPopupImages(img, group = "pnt")
}
```

---

popupGraph   *Create HTML strings for popups*

Description

Create HTML strings for popup graphs used as input for `mapview` or `leaflet`.

Create HTML strings for popup images used as input for `mapview` or `leaflet`.

Create HTML strings for popup tables used as input for `mapview` or `leaflet`. This optionally allows the user to include only a subset of feature attributes.

Usage

```
popupGraph(
  graphs,
  type = c("png", "svg", "html"),
  width = 300,
  height = 300,
  ...
)

popupImage(img, src = c("local", "remote"), embed = FALSE, ...)

popupTable(x, zcol, row.numbers = TRUE, feature.id = TRUE)
```

Arguments

graphs	A `list` of figures associated with x.
type	Output filetype, one of "png" (default), "svg" or "html".
width	popup width in pixels.
height	popup height in pixels.
...	further arguments passed on to underlying methods such as height and width.
img	A character `vector` of file path(s) or web-URL(s) to any sort of image file(s).
src	Whether the source is "local" (i.e. valid file path(s)) or "remote" (i.e. valid URL(s)).
embed	whether to embed the (local) images in the popup html as base64 ecoded. Set this to TRUE if you want to save and share your map, unless you want render many images, then set to FALSE and make sure to copy ../graphs when copying the map to a different location.
x	A `Spatial*` object.
zcol	`numeric` or `character` vector indicating the columns included in the output popup table. If missing, all columns are displayed.
row.numbers	`logical` whether to include row numbers in the popup table.
feature.id	`logical` whether to add 'Feature ID' entry to popup table.

Details

Type `svg` uses native `svg` encoding via [readLines](#). `height` and `width` are set via `...` and passed on to [svg](#)

Type png embeds via "<img src = ...". height and width are set via ... and passed on to png
Type html embeds via "<iframe src = ...". height and width are set directly in pixels.

Value

A list of HTML strings required to create popup graphs.

A list of HTML strings required to create popup graphs.

A list of HTML strings required to create feature popup tables.

Examples

```
if (interactive()) {
### example: svg -----

library(sp)
library(lattice)

data(meuse)
coordinates(meuse) = ~ x + y
proj4string(meuse) = CRS("+init=epsg:28992")
meuse = spTransform(meuse, CRS("+init=epsg:4326"))

## create plots with points colored according to feature id
library(lattice)
p = xyplot(copper ~ cadmium, data = meuse@data, col = "grey")
p = mget(rep("p", length(meuse)))

clr = rep("grey", length(meuse))
p = lapply(1:length(p), function(i) {
  clr[i] = "red"
  update(p[[i]], col = clr)
})

leaflet() %>%
  addTiles() %>%
  addCircleMarkers(data = meuse, popup = popupGraph(p, type = "svg"))

### example: png -----
pt = data.frame(x = 174.764474, y = -36.877245)

coordinates(pt) = ~ x + y
proj4string(pt) = "+init=epsg:4326"

p2 = levelplot(t(volcano), col.regions = terrain.colors(100))

leaflet() %>%
  addTiles() %>%
  addCircleMarkers(data = pt, popup = popupGraph(p2, width = 300, height = 400))

### example: html -----
leaflet() %>%
```

```
    addTiles() %>%
    addCircleMarkers(
      data = breweries91[1, ],
      popup = popupGraph(
        leaflet() %>%
          addProviderTiles("Esri.WorldImagery") %>%
          addMarkers(data = breweries91[1, ],
                     popup = popupTable(breweries91[1, ])),
       type = "html"
      )
    )

}

if (interactive()) {
## remote images -----
### one image
library(sf)

pnt = st_as_sf(data.frame(x = 174.764474, y = -36.877245),
               coords = c("x", "y"),
               crs = 4326)

img = "http://bit.ly/1TVwRiR"

leaflet() %>%
  addTiles() %>%
  addCircleMarkers(data = pnt, popup = popupImage(img, src = "remote"))

### multiple file (types)
library(sp)
images = c(img,
          "https://upload.wikimedia.org/wikipedia/commons/9/91/Octicons-mark-github.svg",
            "https://www.r-project.org/logo/Rlogo.png",
            "https://upload.wikimedia.org/wikipedia/commons/d/d6/MeanMonthlyP.gif")

pt4 = data.frame(x = jitter(rep(174.764474, 4), factor = 0.01),
                 y = jitter(rep(-36.877245, 4), factor = 0.01))
coordinates(pt4) = ~ x + y
proj4string(pt4) = "+init=epsg:4326"

leaflet() %>%
  addTiles() %>%
  addMarkers(data = pt4, popup = popupImage(images)) # NOTE the gif animation

## local images -----
pnt = st_as_sf(data.frame(x = 174.764474, y = -36.877245),
               coords = c("x", "y"), crs = 4326)
img = system.file("img","Rlogo.png",package="png")
leaflet() %>%
  addTiles() %>%
  addCircleMarkers(data = pnt, popup = popupImage(img))
}
```

```
library(leaflet)

leaflet() %>% addTiles() %>% addCircleMarkers(data = breweries91)
leaflet() %>%
  addTiles() %>%
  addCircleMarkers(data = breweries91,
                   popup = popupTable(breweries91))
leaflet() %>%
  addTiles() %>%
  addCircleMarkers(data = breweries91,
                   popup = popupTable(breweries91,
                                      zcol = c("brewery", "zipcode", "village"),
                                      feature.id = FALSE,
                                      row.numbers = FALSE))
```

Index