Package 'leafpm'

October 13, 2022

Version 0.1.0

Date 2019-03-09

Title Leaflet Map Plugin for Drawing and Editing

Description A collection of tools for interactive manipulation of (spatial) data
layers on leaflet web maps. Tools include editing of existing layers, creation
of new layers through drawing of shapes (points, lines, polygons), deletion
of shapes as well as cutting holes into existing shapes. Provides control over
options to e.g. prevent self-intersection of polygons and lines or to enable/disable
snapping to align shapes.

License MIT + file LICENSE

URL https://github.com/r-spatial/leafpm

BugReports https://github.com/r-spatial/leafpm/issues

Encoding UTF-8

LazyData true

ByteCompile true

RoxygenNote 6.1.1

Depends R (>= 3.1.0)

Imports dplyr, htmltools (>= 0.3), htmlwidgets, jsonlite, leaflet (>=
2.0.1), sf (>= 0.5-2),

Enhances geojsonio, mapedit, mapview, shiny

Suggests testthat

NeedsCompilation no

Author Kenton Russell [aut, cre],
Tim Appelhans [aut],
Sumit Kumar [ctb] (Leaflet.pm plugin)

Maintainer Kenton Russell <kent.russell@timelyportfolio.com>

Repository CRAN

Date/Publication 2019-03-13 12:10:03 UTC

R **topics documented:**

---

addPmToolbar   *Add 'leaflet-pm' Toolbar to a Leaflet 'htmlwidget'*

Description

Add 'leaflet-pm' Toolbar to a Leaflet 'htmlwidget'

Usage

```
addPmToolbar(map = NULL, targetLayerId = NULL, targetGroup = NULL,
  toolbarOptions = pmToolbarOptions(), drawOptions = pmDrawOptions(),
  editOptions = pmEditOptions(), cutOptions = pmCutOptions())
```

Arguments

map	leaflet map htmlwidget to which the toolbar should be added
targetLayerId	string id of the feature to edit
targetGroup	string name of the group to edit
toolbarOptions	pmToolbarOptions
drawOptions	pmDrawOptions
editOptions	pmEditOptions
cutOptions	pmCutOptions

Value

leaflet htmlwidget

Examples

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

  # quick example
  leaflet() %>%
```

```
      addTiles() %>%
      addPmToolbar()

   # customizing with options
   leaflet() %>%
      addTiles() %>%
      addPmToolbar(
        toolbarOptions = pmToolbarOptions(drawMarker = FALSE, position = "topright"),
        drawOptions = pmDrawOptions(snappable = FALSE, allowSelfIntersection = FALSE),
        editOptions = pmEditOptions(preventMarkerRemoval = TRUE, draggable = FALSE),
        cutOptions = pmCutOptions(snappable = FALSE, allowSelfIntersection = FALSE)
      )

   # demonstrate that leaflet.pm can work with holes
   library(sf)
   library(leaflet)
   library(leafpm)

   outer1 = matrix(c(0,0,10,0,10,10,0,10,0,0),ncol=2, byrow=TRUE)
   hole1 = matrix(c(1,1,1,2,2,2,2,1,1,1),ncol=2, byrow=TRUE)
   hole2 = matrix(c(5,5,5,6,6,6,6,5,5,5),ncol=2, byrow=TRUE)
   outer2 = matrix(c(11,0,11,1,12,1,12,0,11,0),ncol=2, byrow=TRUE)

   pts1 = list(outer1, hole1, hole2)
   pts2 = list(outer2)

   pl1 = st_sf(geom = st_sfc(st_polygon(pts1)))
   pl2 = st_sf(geom = st_sfc(st_polygon(pts2)))

   mpl = st_sf(geom = st_combine(rbind(pl1, pl2)))

   if(requireNamespace("mapview")) {
     mapview::mapview(mpl)@map %>%
       addPmToolbar(targetGroup = "mpl", cutOptions = pmCutOptions(snappable = FALSE))
   } else {
     warning("Please install mapview to run this example", call. = FALSE)
   }

 }
```

---

pmCutOptions	*Set Options for 'leaflet.pm' Cut Mode*

Description

Set Options for 'leaflet.pm' Cut Mode

Usage

```
pmCutOptions(snappable = FALSE, allowSelfIntersection = TRUE,
  cursorMarker = FALSE)
```

Arguments

`snappable`	logical to snap while drawing. Default is `TRUE`.
`allowSelfIntersection`	
	logical to allow self-intersection. Default is `TRUE`.
`cursorMarker`	logical to show a marker at the cursor. Default is `TRUE`.

Value

list of options

`pmDependencies`	*Dependencies for 'leaflet-pm'*

Description

Dependencies for 'leaflet-pm'

Usage

```
pmDependencies()
```

Value

htmltools::htmlDependency

`pmDrawOptions`	*Set Options for 'leaflet.pm' Draw Mode*

Description

Set Options for 'leaflet.pm' Draw Mode

Usage

```
pmDrawOptions(snappable = TRUE, snapDistance = 20, snapMiddle = TRUE,
  tooltips = TRUE, cursorMarker = TRUE, finishOn = NULL,
  allowSelfIntersection = TRUE, templineStyle = list(),
  hintlineStyle = list(color = "#3388ff", dashArray = "5,5"),
  markerStyle = list(draggable = TRUE))
```

Arguments

`snappable`	logical to snap while drawing. Default is `TRUE`.
`snapDistance`	integer for the distance within which snapping will occur. Default is `20`.
`snapMiddle`	logical to snap in the middle of segments. Default is `TRUE`.
`tooltips`	logical to show tooltips. Default is `TRUE`.
`cursorMarker`	logical to show a marker at the cursor. Default is `TRUE`.
`finishOn`	string type of layer event to finish the drawn shape. Example events are 'mouseout', 'dblclick', 'contextmenu'. Default is `NULL`.
`allowSelfIntersection`	
	logical to allow self-intersection. Default is `TRUE`.
`templineStyle`	list to customize the styling of the lines between coordinates and markers.
`hintlineStyle`	list to customize the styling of the line between the last marker and the cursor.
`markerStyle`	list to customize the styling of the added marker.

Value

list of options

`pmEditOptions`	*Set Options for 'leaflet.pm' Edit Mode*

Description

Set Options for 'leaflet.pm' Edit Mode

Usage

```
pmEditOptions(snappable = TRUE, snapDistance = 20,
  allowSelfIntersection = TRUE, draggable = TRUE,
  preventMarkerRemoval = FALSE, preventVertexEdit = FALSE)
```

Arguments

`snappable`	logical to snap while drawing. Default is `TRUE`.
`snapDistance`	integer for the distance within which snapping will occur. Default is `20`.
`allowSelfIntersection`	
	logical to allow self-intersection. Default is `TRUE`.
`draggable`	logical make the layer draggable. Default is `TRUE`.
`preventMarkerRemoval`	
	logical to disable removal via right-click. Default is `FALSE`.
`preventVertexEdit`	
	logical to disable editing of vertices. Default is `FALSE`.

Value

list of options

---

pmToolbarOptions   *Set Options for 'leaflet.pm' Toolbar*

Description

Set Options for 'leaflet.pm' Toolbar

Usage

```
pmToolbarOptions(drawMarker = TRUE, drawPolygon = TRUE,
  drawPolyline = TRUE, drawCircle = TRUE, drawRectangle = TRUE,
  editMode = TRUE, cutPolygon = TRUE, removalMode = TRUE,
  position = "topleft")
```

Arguments

drawMarker	`logical` add button to draw a marker. Default is TRUE.
drawPolygon	`logical` add button to draw a polygon. Default is TRUE.
drawPolyline	`logical` add button to draw a polyline. Default is TRUE.
drawCircle	`logical` add button to draw a circle. Default is TRUE.
drawRectangle	`logical` add button to draw a rectangle. Default is TRUE.
editMode	`logical` add button to edit features. Default is TRUE.
cutPolygon	`logical` add button to cut a hole. Default is TRUE.
removalMode	`logical` add button to remove features. Default is TRUE.
position	`string` toolbar position. Options are 'topleft', 'topright', 'bottomleft', 'bottomright'. Default is `'topleft'`.

Value

`leaflet` htmlwidget with added toolbar

---

removePmToolbar   *Removes the 'leaflet.pm' toolbar*

Description

Removes the 'leaflet.pm' toolbar

Usage

```
removePmToolbar(map, clearFeatures = FALSE)
```

Arguments

`map`	`leaflet` map htmlwidget from which the toolbar should be removed
`clearFeatures`	whether to clear the map of drawn features. currently not working

Value

`leaflet` htmlwidget

Index