

Package ‘leafletlegend’

May 9, 2024

Type Package

Title Add Custom Legends to 'leaflet' Maps

Version 1.2.1

Description Provides extensions to the 'leaflet' package to customize legends with images, text styling, orientation, sizing, and symbology and functions to create symbols to plot on maps.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 3.3.0)

Imports leaflet, htmltools, stats, base64enc, htmlwidgets

RoxygenNote 7.2.3

URL <https://leafletlegend.delveds.com>,
<https://github.com/tomroh/leafletlegend>

BugReports <https://github.com/tomroh/leafletlegend/issues>

Suggests covr, testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Thomas Roh [aut, cre],
Ricardo Rodrigo Basa [ctb]

Maintainer Thomas Roh <thomas@roh.engineering>

Repository CRAN

Date/Publication 2024-05-09 15:00:03 UTC

R topics documented:

| | |
|--------------------------------|----|
| addLeafLegends | 2 |
| addLegendAwesomeIcon | 7 |
| addLegendImage | 9 |
| availableShapes | 11 |
| legendSymbols | 11 |
| mapSymbols | 16 |

| | |
|----------------|---|
| addLeafLegends | <i>Add Customizable Color Legends to a 'leaflet' map widget</i> |
|----------------|---|

Description

Functions for more control over the styling of 'leaflet' legends. The 'leaflet' map is passed through and the output is a 'leaflet' control so that the legends are integrated with other functionality of the API. Style the text of the labels, the symbols used, orientation of the legend items, and sizing of all elements.

Usage

```
addLegendNumeric(  
  map,  
  pal,  
  values,  
  title = NULL,  
  shape = c("rect", "stadium"),  
  orientation = c("vertical", "horizontal"),  
  width = 20,  
  height = 100,  
  bins = 7,  
  numberFormat = function(x) {  
    prettyNum(x, format = "f", big.mark = ",", digits =  
      3, scientific = FALSE)  
  },  
  tickLength = 4,  
  tickWidth = 1,  
  decreasing = FALSE,  
  fillOpacity = 1,  
  group = NULL,  
  labels = NULL,  
  naLabel = "NA",  
  labelStyle = "",  
  className = "info legend leaflet-control",  
  data = leaflet::getMapData(map),  
  ...  
)  
  
addLegendQuantile(  
  map,  
  pal,  
  values,  
  title = NULL,  
  labelStyle = "",  
  shape = "rect",
```

```
orientation = c("vertical", "horizontal"),
width = 24,
height = 24,
numberFormat = function(x) {
  prettyNum(x, big.mark = ",", scientific = FALSE,
  digits = 1)
},
opacity = 1,
fillOpacity = opacity,
group = NULL,
className = "info legend leaflet-control",
naLabel = "NA",
between = " - ",
data = leaflet::getMapData(map),
...
)

addLegendBin(
  map,
  pal,
  values,
  title = NULL,
  labelStyle = "",
  shape = "rect",
  orientation = c("vertical", "horizontal"),
  width = 24,
  height = 24,
  numberFormat = function(x) {
    format(round(x, 3), big.mark = ",", trim = TRUE,
    scientific = FALSE)
  },
  opacity = 1,
  fillOpacity = opacity,
  group = NULL,
  className = "info legend leaflet-control",
  naLabel = "NA",
  between = " - ",
  data = leaflet::getMapData(map),
  ...
)

addLegendFactor(
  map,
  pal,
  values,
  title = NULL,
  labelStyle = "",
  shape = "rect",
```

```

orientation = c("vertical", "horizontal"),
width = 24,
height = 24,
opacity = 1,
fillOpacity = opacity,
group = NULL,
className = "info legend leaflet-control",
naLabel = "NA",
data = leaflet::getMapData(map),
...
)

```

Arguments

| | |
|--------------|---|
| map | a map widget object created from 'leaflet' |
| pal | the color palette function, generated from colorNumeric |
| values | the values used to generate colors from the palette function |
| title | the legend title, pass in HTML to style |
| shape | the desired shape of the symbol, See availableShapes |
| orientation | stack the legend items vertically or horizontally |
| width | in pixels |
| height | in pixels |
| bins | an approximate number of tick-marks on the color gradient for the colorNumeric palette if it is of length one; you can also provide a numeric vector as the pre-defined breaks |
| numberFormat | formatting functions for numbers that are displayed e.g. format, prettyNum |
| tickLength | in pixels |
| tickWidth | in pixels |
| decreasing | order of numbers in the legend |
| fillOpacity | fill opacity of the legend items |
| group | group name of a leaflet layer group |
| labels | labels |
| naLabel | the legend label for NAs in values |
| labelStyle | character string of style argument for HTML text |
| className | extra CSS class to append to the control, space separated |
| data | a data object. Currently supported objects are matrices, data frames, spatial objects from the sp package (SpatialPoints, SpatialPointsDataFrame, Polygon, Polygons, SpatialPolygons, SpatialPolygonsDataFrame, Line, Lines, SpatialLines, and SpatialLinesDataFrame), and spatial data frames from the sf package. |
| ... | arguments to pass to addControl |
| opacity | opacity of the legend items |
| between | a separator between legend range labels |

Value

an object from [addControl](#)

Examples

```
library(leaflet)

data(quakes)

# Numeric Legend

numPal <- colorNumeric('viridis', quakes$depth)
leaflet() %>%
  addTiles() %>%
  addLegendNumeric(
    pal = numPal,
    values = quakes$depth,
    position = 'topright',
    title = 'addLegendNumeric (Horizontal)',
    orientation = 'horizontal',
    shape = 'rect',
    decreasing = FALSE,
    height = 20,
    width = 100
  ) %>%
  addLegendNumeric(
    pal = numPal,
    values = quakes$depth,
    position = 'topright',
    title = htmltools::tags$div('addLegendNumeric (Decreasing)',
    style = 'font-size: 24px; text-align: center; margin-bottom: 5px;'),
    orientation = 'vertical',
    shape = 'stadium',
    decreasing = TRUE,
    height = 100,
    width = 20
  ) %>%
  addLegend(pal = numPal, values = quakes$depth, title = 'addLegend')

# Quantile Legend
# defaults to adding quantile numeric break points

quantPal <- colorQuantile('viridis', quakes$mag, n = 5)
leaflet() %>%
  addTiles() %>%
  addCircleMarkers(data = quakes,
    lat = ~lat,
    lng = ~long,
    color = ~quantPal(mag),
    opacity = 1,
    fillOpacity = 1
  ) %>%
```

```

addLegendQuantile(pal = quantPal,
                  values = quakes$mag,
                  position = 'topright',
                  title = 'addLegendQuantile',
                  numberFormat = function(x) {prettyNum(x, big.mark = ',',
                  scientific = FALSE, digits = 2)},
                  shape = 'circle') %>%
addLegendQuantile(pal = quantPal,
                  values = quakes$mag,
                  position = 'topright',
                  title = htmltools::tags$div('addLegendQuantile',
                  htmltools::tags$br(),
                  '(Omit Numbers)'),
                  numberFormat = NULL,
                  shape = 'circle') %>%
addLegend(pal = quantPal, values = quakes$mag, title = 'addLegend')

# Factor Legend
# Style the title with html tags, several shapes are supported drawn with svg

quakes[['group']] <- sample(c('A', 'B', 'C'), nrow(quakes), replace = TRUE)
factorPal <- colorFactor('Dark2', quakes$group)
leaflet() %>%
  addTiles() %>%
  addCircleMarkers(
    data = quakes,
    lat = ~ lat,
    lng = ~ long,
    color = ~ factorPal(group),
    opacity = 1,
    fillOpacity = 1
  ) %>%
  addLegendFactor(
    pal = factorPal,
    title = htmltools::tags$div('addLegendFactor', style = 'font-size: 24px;
    color: red;'),
    values = quakes$group,
    position = 'topright',
    shape = 'triangle',
    width = 50,
    height = 50
  ) %>%
  addLegend(pal = factorPal,
            values = quakes$group,
            title = 'addLegend')

# Bin Legend
# Restyle the text of the labels, change the legend item orientation

binPal <- colorBin('Set1', quakes$mag)
leaflet(quakes) %>%
  addTiles() %>%
  addCircleMarkers(

```

```

    lat = ~ lat,
    lng = ~ long,
    color = ~ binPal(mag),
    opacity = 1,
    fillOpacity = 1
  ) %>%
  addLegendBin(
    pal = binPal,
    position = 'topright',
    values = ~mag,
    title = 'addLegendBin',
    labelStyle = 'font-size: 18px; font-weight: bold;',
    orientation = 'horizontal'
  ) %>%
  addLegend(pal = binPal,
            values = quakes$mag,
            title = 'addLegend')

# Group Layer Control
# Works with baseGroups and overlayGroups

leaflet() %>%
  addTiles() %>%
  addLegendNumeric(
    pal = numPal,
    values = quakes$depth,
    position = 'topright',
    title = 'addLegendNumeric',
    group = 'Numeric Data'
  ) %>%
  addLegendQuantile(
    pal = quantPal,
    values = quakes$mag,
    position = 'topright',
    title = 'addLegendQuantile',
    group = 'Quantile'
  ) %>%
  addLegendBin(
    data = quakes,
    pal = binPal,
    position = 'bottomleft',
    title = 'addLegendBin',
    group = 'Bin',
    values = ~mag
  ) %>%
  addLayersControl(
    baseGroups = c('Numeric Data', 'Quantile'), overlayGroups = c('Bin'),
    position = 'bottomright'
  )

```

Description

Add a legend with Awesome Icons

Usage

```
addLegendAwesomeIcon(
  map,
  iconSet,
  title = NULL,
  labelStyle = "",
  orientation = c("vertical", "horizontal"),
  marker = TRUE,
  group = NULL,
  className = "info legend leaflet-control",
  ...
)
```

Arguments

| | |
|-------------|--|
| map | a map widget object created from 'leaflet' |
| iconSet | a named list from awesomeIconList , the names will be the labels in the legend |
| title | the legend title, pass in HTML to style |
| labelStyle | character string of style argument for HTML text |
| orientation | stack the legend items vertically or horizontally |
| marker | whether to show the marker or only the icon |
| group | group name of a leaflet layer group |
| className | extra CSS class to append to the control, space separated |
| ... | arguments to pass to addControl |

Value

an object from [addControl](#)

Examples

```
library(leaflet)
data(quakes)
iconSet <- awesomeIconList(
  `Font Awesome` = makeAwesomeIcon(icon = "font-awesome", library = "fa",
    iconColor = 'gold', markerColor = 'red',
    spin = FALSE,
    squareMarker = TRUE,
    iconRotate = 30,
  ),
  Ionic = makeAwesomeIcon(icon = "ionic", library = "ion",
    iconColor = '#ffffff', markerColor = 'blue',
    spin = TRUE,
```



```

        squareMarker = FALSE),
  Glyphicon = makeAwesomeIcon(icon = "plus-sign", library = "glyphicon",
                              iconColor = 'rgb(192, 255, 0)',
                              markerColor = 'darkpurple',
                              spin = TRUE,
                              squareMarker = FALSE)
)
leaflet(quakes[1:3,]) %>%
  addTiles() %>%
  addAwesomeMarkers(lat = ~lat,
                    lng = ~long,
                    icon = iconSet) %>%
  addLegendAwesomeIcon(iconSet = iconSet,
                       orientation = 'horizontal',
                       title = htmltools::tags$div(
                         style = 'font-size: 20px;',
                         'Awesome Icons'),
                       labelStyle = 'font-size: 16px;') %>%
  addLegendAwesomeIcon(iconSet = iconSet,
                       orientation = 'vertical',
                       marker = FALSE,
                       title = htmltools::tags$div(
                         style = 'font-size: 20px;',
                         'Awesome Icons'),
                       labelStyle = 'font-size: 16px;')

```

addLegendImage

Add a Legend with Images

Description

Creates a legend with images that are embedded into a 'leaflet' map so that images do not need to be packaged when saving a 'leaflet' map as HTML. Full control over the label and title style. The 'leaflet' map is passed through and the output is a control so that legend is fully integrated with other functionalities.

Usage

```

addLegendImage(
  map,
  images,
  labels,
  title = NULL,
  labelStyle = "font-size: 24px; vertical-align: middle;",
  orientation = c("vertical", "horizontal"),
  width = 20,
  height = 20,
  group = NULL,
  className = "info legend leaflet-control",

```

```
    ...
  )
```

Arguments

| | |
|-------------|---|
| map | a map widget object created from 'leaflet' |
| images | path to the image file |
| labels | labels for each image |
| title | the legend title, pass in HTML to style |
| labelStyle | character string of style argument for HTML text |
| orientation | stack the legend items vertically or horizontally |
| width | in pixels |
| height | in pixels |
| group | group name of a leaflet layer group |
| className | extra CSS class to append to the control, space separated |
| ... | arguments to pass to addControl |

Value

an object from [addControl](#)

Examples

```
library(leaflet)
data(quakes)

quakes1 <- quakes[1:10,]

colors <- c('blue', 'red', 'yellow', 'green', 'orange', 'purple')
i <- as.integer(cut(quakes$mag, breaks = quantile(quakes$mag, seq(0,1,1/6)),
  include.lowest = TRUE))
leafImg <- system.file(sprintf('img/leaf-%s.png', colors),
  package = 'leafletlegend')
leafIcons <- icons(
  iconUrl = leafImg[i],
  iconWidth = 133/236 * 50, iconHeight = 50
)
leaflet(data = quakes) %>% addTiles() %>%
  addMarkers(~long, ~lat, icon = leafIcons) %>%
  addLegendImage(images = leafImg,
    labels = colors,
    width = 133/236 * 50,
    height = 50,
    orientation = 'vertical',
    title = htmltools::tags$div('Leaf',
      style = 'font-size: 24px;
      text-align: center;'),
```

```

        position = 'topright')

# use raster images with size encodings
height <- sizeNumeric(quakes$depth, baseSize = 40)
width <- height * 38 / 95
symbols <- icons(
  iconUrl = leafImg[4],
  iconWidth = width,
  iconHeight = height)
probs <- c(.2, .4, .6, .8)
leaflet(quakes) %>%
  addTiles() %>%
  addMarkers(icon = symbols,
             lat = ~lat, lng = ~long) %>%
  addLegendImage(
    images = rep(leafImg[4], 4),
    labels = round(quantile(height, probs = probs), 0),
    width = quantile(height, probs = probs) * 38 / 95,
    height = quantile(height, probs = probs),
    title = htmltools::tags$div(
      'Leaf',
      style = 'font-size: 24px; text-align: center; margin-bottom: 5px;'),
    position = 'topright', orientation = 'vertical')

```

| | |
|-----------------|---|
| availableShapes | <i>Available shapes for map symbols</i> |
|-----------------|---|

Description

Available shapes for map symbols

Usage

```
availableShapes()
```

Value

list of available shapes

| | |
|---------------|---|
| legendSymbols | <i>Add a legend for the sizing of symbols or the width of lines</i> |
|---------------|---|

Description

Add a legend for the sizing of symbols or the width of lines

Usage

```
addLegendSize(  
  map,  
  pal,  
  values,  
  title = NULL,  
  labelStyle = "",  
  shape = "rect",  
  orientation = c("vertical", "horizontal"),  
  color,  
  fillColor = color,  
  strokeWidth = 1,  
  opacity = 1,  
  fillOpacity = opacity,  
  breaks = 5,  
  baseSize = 20,  
  numberFormat = function(x) {  
    prettyNum(x, big.mark = ",", scientific = FALSE,  
              digits = 1)  
  },  
  group = NULL,  
  className = "info legend leaflet-control",  
  stacked = FALSE,  
  data = leaflet::getMapData(map),  
  ...  
)  
  
addLegendLine(  
  map,  
  pal,  
  values,  
  title = NULL,  
  labelStyle = "",  
  orientation = c("vertical", "horizontal"),  
  width = 20,  
  color,  
  opacity = 1,  
  fillOpacity = opacity,  
  breaks = 5,  
  baseSize = 10,  
  numberFormat = function(x) {  
    prettyNum(x, big.mark = ",", scientific = FALSE,  
              digits = 1)  
  },  
  group = NULL,  
  className = "info legend leaflet-control",  
  data = leaflet::getMapData(map),  
  ...  
)
```

```

)

addLegendSymbol(
  map,
  pal,
  values,
  title = NULL,
  labelStyle = "",
  shape,
  orientation = c("vertical", "horizontal"),
  color,
  fillColor = color,
  strokeWidth = 1,
  opacity = 1,
  fillOpacity = opacity,
  width = 20,
  height = width,
  group = NULL,
  className = "info legend leaflet-control",
  dashArray = NULL,
  data = leaflet::getMapData(map),
  ...
)

```

Arguments

| | |
|--------------|---|
| map | a map widget object created from 'leaflet' |
| pal | the color palette function, generated from colorNumeric |
| values | the values used to generate sizes and if colorValues is not specified and pal is given, then the values are used to generate colors from the palette function |
| title | the legend title, pass in HTML to style |
| labelStyle | character string of style argument for HTML text |
| shape | the desired shape of the symbol, See availableShapes |
| orientation | stack the legend items vertically or horizontally |
| color | the color of the legend symbols, if omitted pal is used |
| fillColor | fill color of symbol |
| strokeWidth | width of symbol outline |
| opacity | opacity of the legend items |
| fillOpacity | fill opacity of the legend items |
| breaks | an integer specifying the number of breaks or a numeric vector of the breaks |
| baseSize | re-scaling size in pixels of the mean of the values, the average value will be this exact size |
| numberFormat | formatting functions for numbers that are displayed e.g. format, prettyNum |
| group | group name of a leaflet layer group |

| | |
|-----------|---|
| className | extra CSS class to append to the control, space separated |
| stacked | If TRUE, symbols are overlaid onto each other for a more compact size legend |
| data | a data object. Currently supported objects are matrices, data frames, spatial objects from the sp package (SpatialPoints, SpatialPointsDataFrame, Polygon, Polygons, SpatialPolygons, SpatialPolygonsDataFrame, Line, Lines, SpatialLines, and SpatialLinesDataFrame), and spatial data frames from the sf package. |
| ... | arguments to pass to addControl for addLegendSize pretty for sizeBreaks makeSymbol for makeSymbolsSize |
| width | width in pixels of the lines |
| height | in pixels |
| dashArray | a string or vector/list of strings that defines the stroke dash pattern |

Value

an object from [addControl](#)

Examples

```
library(leaflet)
data("quakes")
quakes <- quakes[1:100,]
numPal <- colorNumeric('viridis', quakes$depth)
sizes <- sizeNumeric(quakes$depth, baseSize = 10)
symbols <- Map(
  makeSymbol,
  shape = 'triangle',
  color = numPal(quakes$depth),
  width = sizes,
  height = sizes
)
leaflet() %>%
  addTiles() %>%
  addMarkers(data = quakes,
             icon = icons(iconUrl = symbols),
             lat = ~lat, lng = ~long) %>%
  addLegendSize(
    values = quakes$depth,
    pal = numPal,
    title = 'Depth',
    labelStyle = 'margin: auto;',
    shape = c('triangle'),
    orientation = c('vertical', 'horizontal'),
    opacity = .7,
    breaks = 5)

# a wrapper for making icons is provided
sizeSymbols <-
```

```

makeSymbolsSize(
  quakes$depth,
  shape = 'cross',
  fillColor = numPal(quakes$depth),
  color = 'black',
  strokeWidth = 1,
  opacity = .8,
  fillOpacity = .5,
  baseSize = 20
)
leaflet() %>%
  addTiles() %>%
  addMarkers(data = quakes,
             icon = sizeSymbols,
             lat = ~lat, lng = ~long) %>%
  addLegendSize(
    values = quakes$depth,
    pal = numPal,
    title = 'Depth',
    shape = 'cross',
    orientation = 'horizontal',
    strokeWidth = 1,
    opacity = .8,
    fillOpacity = .5,
    color = 'black',
    baseSize = 20,
    breaks = 5)

# Group layers control
leaflet() %>%
  addTiles() %>%
  addLegendSize(
    values = quakes$depth,
    pal = numPal,
    title = 'Depth',
    labelStyle = 'margin: auto;',
    shape = c('triangle'),
    orientation = c('vertical', 'horizontal'),
    opacity = .7,
    breaks = 5,
    group = 'Depth') %>%
  addLayersControl(overlayGroups = c('Depth'))

# Polyline Legend for Size
baseSize <- 10
lineColor <- '#00000080'
pal <- colorNumeric('Reds', at1Storms2005$MinPress)
leaflet() %>%
  addTiles() %>%
  addPolylines(data = at1Storms2005,
              weight = ~sizeNumeric(values = MaxWind, baseSize = baseSize),
              color = ~pal(MinPress),
              popup = ~as.character(MaxWind)) %>%

```

```

addLegendLine(values = atlStorms2005$MaxWind,
              title = 'MaxWind',
              baseSize = baseSize,
              width = 50,
              color = lineColor) %>%
addLegendNumeric(pal = pal,
                 title = 'MinPress',
                 values = atlStorms2005$MinPress)

# Stacked Legends
leaflet(quakes) %>%
addTiles() %>%
  addSymbolsSize(values = ~10^(mag),
                 lat = ~lat,
                 lng = ~long,
                 shape = 'circle',
                 color = 'black',
                 fillColor = 'red',
                 opacity = 1,
                 baseSize = 5) |>
addLegendSize(
  values = ~10^(mag),
  title = 'Magnitude',
  baseSize = 5,
  shape = 'circle',
  color = 'black',
  fillColor = 'red',
  labelStyle = 'font-size: 18px;',
  position = 'bottomleft',
  stacked = TRUE,
  breaks = 5)

```

mapSymbols

Create Map Symbols for 'leaflet' maps

Description

Create Map Symbols for 'leaflet' maps

Usage

```

makeSymbol(
  shape,
  width,
  height = width,
  color,
  fillColor = color,
  opacity = 1,
  fillOpacity = opacity,

```



```
    ...
  )

  makeSvgUri(svg, width, height, strokeWidth)

  makeSymbolIcons(
    shape,
    color,
    fillColor = color,
    opacity,
    fillOpacity = opacity,
    strokeWidth = 1,
    width,
    height = width,
    ...
  )

  addSymbols(
    map,
    lng,
    lat,
    values,
    shape,
    color,
    fillColor = color,
    opacity = 1,
    fillOpacity = opacity,
    strokeWidth = 1,
    width = 20,
    height = width,
    dashArray = NULL,
    data = leaflet::getMapData(map),
    ...
  )

  addSymbolsSize(
    map,
    lng,
    lat,
    values,
    shape,
    color,
    fillColor = color,
    opacity = 1,
    fillOpacity = opacity,
    strokeWidth = 1,
    baseSize = 20,
    data = leaflet::getMapData(map),
```

```

    ...
  )

sizeNumeric(values, baseSize)

sizeBreaks(values, breaks, baseSize, ...)

makeSymbolsSize(
  values,
  shape = "rect",
  color,
  fillColor,
  opacity = 1,
  fillOpacity = opacity,
  strokeWidth = 1,
  baseSize,
  ...
)

```

Arguments

| | |
|-------------|--|
| shape | the desired shape of the symbol, See availableShapes |
| width | in pixels |
| height | in pixels |
| color | stroke color |
| fillColor | fill color |
| opacity | stroke opacity |
| fillOpacity | fill opacity |
| ... | arguments to pass to pretty |
| svg | inner svg tags for symbol |
| strokeWidth | stroke width in pixels |
| map | a map widget object created from 'leaflet' |
| lng | a numeric vector of longitudes, or a one-sided formula of the form $\sim x$ where x is a variable in data; by default (if not explicitly provided), it will be automatically inferred from data by looking for a column named lng, long, or longitude (case-insensitively) |
| lat | a vector of latitudes or a formula (similar to the lng argument; the names lat and latitude are used when guessing the latitude column from data) |
| values | the values used to generate shapes; can be omitted for a single type of shape |
| dashArray | a string or vector/list of strings that defines the stroke dash pattern |
| data | the data object from which the argument values are derived; by default, it is the data object provided to <code>leaflet()</code> initially, but can be overridden |
| baseSize | re-scaling size in pixels of the mean of the values, the average value will be this exact size |
| breaks | an integer specifying the number of breaks or a numeric vector of the breaks; if a named vector then the names are used as labels. |

Value

HTML svg element

Index

`addControl`, [4](#), [5](#), [8](#), [10](#), [14](#)
`addLeafLegends`, [2](#)
`addLegendAwesomeIcon`, [7](#)
`addLegendBin` (`addLeafLegends`), [2](#)
`addLegendFactor` (`addLeafLegends`), [2](#)
`addLegendImage`, [9](#)
`addLegendLine` (`legendSymbols`), [11](#)
`addLegendNumeric` (`addLeafLegends`), [2](#)
`addLegendQuantile` (`addLeafLegends`), [2](#)
`addLegendSize` (`legendSymbols`), [11](#)
`addLegendSymbol` (`legendSymbols`), [11](#)
`addSymbols` (`mapSymbols`), [16](#)
`addSymbolsSize` (`mapSymbols`), [16](#)
`availableShapes`, [4](#), [11](#), [13](#), [18](#)
`awesomeIconList`, [8](#)

`colorNumeric`, [4](#), [13](#)

`legendSymbols`, [11](#)

`makeSvgUri` (`mapSymbols`), [16](#)
`makeSymbol`, [14](#)
`makeSymbol` (`mapSymbols`), [16](#)
`makeSymbolIcons` (`mapSymbols`), [16](#)
`makeSymbolsSize` (`mapSymbols`), [16](#)
`mapSymbols`, [16](#)

`pretty`, [14](#)

`sizeBreaks` (`mapSymbols`), [16](#)
`sizeNumeric` (`mapSymbols`), [16](#)