Package 'ROpenLayers'

February 3, 2019

Title Geo-visualization Using OperLayers and ArcGIS (SIPR)

Version 1.0.0-99

Description Functions to export geospatial data and analyses to interactive HTML/javascript visualization using the OpenLayers javascript library. The resulting HTML pages replicate some of the functionality in the Leaflet and ggmap packages, but have the advantage of enabling the user to easily leverage the functionality of the OpenLayers javascript library with NGA map servers. User-supplied ArcGIS map servers are also supported. The output HTML, files, and folders can be viewed on a local machine, hosted as self-contained web pages on a minimal http server, or parsed by the user for inclusion in other applications, web pages, or other server environments (e.g., RShiny).

Depends R (>= 3.3)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports sp, stats, grDevices, graphics, utils, base64enc, png

Suggests rgdal, jpeg, tiff, httr, jsonlite

RoxygenNote 6.1.1

NeedsCompilation no

Author Christopher Marks [aut, cre]

Maintainer Christopher Marks <cemarks@alum.mit.edu>

R topics documented:

+.Ol.Map *+.Ol.Map*

Description

Add components to a OpenLayers Map.

Usage

```
## S3 method for class 'Ol.Map'
ol.map.obj + other.obj
```

Arguments

`ol.map.obj`	S3 object of class `Ol.Map`.
`other.obj`	A map layer or scale component.

Details

Similar to the `ggplot2` package, + provides functionality to add layers to an existing OpenLayers Map object. Layers are simply appended to the Ol.Map objects layers list. When adding scales, this method searches through map layers in reverse order for scales with matching aesthetics. When a matching scale is found, it is updated according to the parameters of the added scale. In general, continuous scales can be coerced into discrete scales.

Value

Ol.Map object with updated layers or scales.

What can you add?

You can add the following types of objects:

- A layer object generated by one of the `ol_geom_*` layer functions.
- A scale object generated by one of the `ol_scale_*` functions.

See Also

ol_map

Examples

```
mymap <- ol_map()
base.layer <- lightgray()
mymap <- mymap + base.layer
## Not run:
ol_map2HTML(mymap,"SanDiego.html")
browseURL("SanDiego.html")

## End(Not run)
```

---

geocode *Geocode an Address*

Description

Get Lat/Lon Cooridinates for an Address.

Usage

```
geocode(address.strings)
```

Arguments

address.strings
character vector of addresses to geocode.

Details

This function uses the geocoding utility at NGA.smil.mil to get coordinates for an address or list of addresses.

Value

A data frame with a `location` column providing latitude and longitude values for each geo-coded location.

See Also

ol_geom_point, ol_geom_icon,

Examples

```
addresses <- c(
    "1600 Pennsylvania Ave NW, Washington, DC 20500",
    "One 1st ST NE, Washington, DC  20543"
)
## Not run:
g <- geocode(addresses)
point.matrix <- g$location
point.df <- data.frame(
    pt.type=c("White House","Supreme Court")
)
mymap <- ol_map(
    center=c(-77.03196,38.89037),
   zoom=12
) +
    streetmap() +
ol_geom_point(
    point.matrix,
    name="Points of Interest",
    marker="pin",
    toggle.control=TRUE,
    tooltip=point.df$pt.type
)

# Output to file and view
# ol_map2HTML(mymap,'map.html')
# browseURL('map.html')

## End(Not run)
```

---

nga_basemap *NGA Basemap Layer*

Description

Create a basemap layer linking to an NGA ArcGIS mapserver.

Usage

```
nga_basemap(basemap.identifier = "WSM", name = NULL,
  toggle.control = FALSE)

streetmap(toggle.control = FALSE)

lightgray(toggle.control = FALSE)

oceanbase(toggle.control = FALSE)
```

Arguments

`basemap.identifier`
character indicating which NGA mapserver to use. See 'Available Base Maps'.

`name` character layer name.

`toggle.control` logical. If `TRUE`, a checkbox will appear on the map allowing the viewer to toggle its visibility in the browser.

Details

Creates and returns an OpenLayers ArcGIS Tile layer that sources a map server hosted at `http://home.gvs.nga.smil.mil`. These map servers are owned by the US Government and require authentication. If the `basemap.identifier` parameter is unrecognized the function will default to the NGA OpenStreetMap map server.

Value

A `Layer.ArcGIS` S3 object.

Functions

- `streetmap`: Call "WSM" `nga_basemap`
- `lightgray`: Call "LightGray" `nga_basemap`.
- `oceanbase`: Call "LightMidnight" `nga_basemap`.

Available Base Maps

The following basemap.identifiers are currently supported by this method.

"ABM"	Analytic Base Map
"LightGray"	Analytic Base Map (Light Gray)
"Light_LightGray"	Analytic Base Map (Light Light Gray)
"LightMidnight"	Analytic Base Map (Light Midnight)
"Light_Slate"	Analytic Base Map (Light Slate)
"Midnight"	Analytic Base Map (Midnight)
"Slate"	Analytic Base Map (Slate)
"CARDG"	Scanned CARDG Maps
"DNC"	Digital Nautical Charts
"Imagery"	Satellite Imagery
"Hillshade"	Hillshade Map
"ShadedRelief"	Shaded Relief Map
"TintedHillshade"	Tinted Hillshade Map
"WorldBoundaries"	World Boundaries (WSM)
"WorldBoundaries_Places"	World Boundaries, Places (WSM)
"WorldPlaceNames"	World Place Names (WSM)
"WorldTransportation"	World Transportation (WSM)
"WSM"	World Street Map

See Also

`ol_map`, `+.Ol.Map`, `nga_basemap`, `user_arcgis_basemap`

Examples

```
mymap <- ol_map()
base.layer <- nga_basemap('Midnight')
mymap <- mymap + base.layer
## Not run:
ol_map2HTML(mymap,"SanDiegoMidnight.html")
browseURL("SanDiegoMidnight.html")

## End(Not run)
```

`ol_aes` *Aesthetic Mappings*

Description

Map variables to layer aesthetics.

Usage

```
ol_aes(...)
```

Arguments

`...`	comma-separated mappings of the form 'aesthetic=variable'. Available aesthetics for mapping are layer specific and are listed in the documentation for each layer type. Unavailable or unrecognized aesthetics are ignored. Variables must correspond to names in the layer's input data.frame, otherwise an error is thrown.

Details

This function replicates a subset of the functionality of the ggplot2 `aes` function. It *does not* allow for variable transformations or functions of multpile varuables. These operations must be completed a priori by the user.

Value

A list of aesthetic mappings.

See Also

`ol_geom_polygon`, `ol_geom_line`, `ol_geom_point`, `ol_geom_icon`, `ol_geom_circle`

Examples

```
polygon.matrix1 <- matrix(
    c(
        -80.385+c(0,0.05,0.05,0,0),
        25.782618+c(0,0,0.05,0.05,0)
    ),
    ncol=2
)
polygon.matrix2 <- matrix(
    c(
        -80.34+c(0,0.05,0.025,0),
        25.73++c(0,0,0.025*sqrt(3),0)
    ),
    ncol=2
)
polygon.list<-list(polygon.matrix1,polygon.matrix2)
polygon.df <- data.frame(shape=c("rectangle","triangle"),no=c(1,2))
miami.OSM.basemap <- ol_map(
    center=c(-80.385790,25.782618),
    zoom=9
    ) +
   streetmap()
polygon.layer <- ol_geom_polygon(
    polygon.list,
    mapping=ol_aes(
        fill=no,
        lwd=shape
    ),
    df=polygon.df,
    name="Miami Polygons",
    toggle.control=TRUE,
    tooltip=polygon.df$no
)
polygon.fill.scale <- ol_scale_fill_discrete(
    c("1"="red","2"="green"),
    opacity=0.5,
    display=TRUE,
    name="Number"
)
polygon.linewidth.scale <- ol_scale_lwd_discrete(
    display=TRUE,
    name="Shape"
)
polygons.over.miami <- miami.OSM.basemap +
    polygon.layer +
    polygon.fill.scale +
    polygon.linewidth.scale

## Not run:
ol_map2HTML(
  polygons.over.miami,
  'miami_polygons.html',
  map.heading="Miami Shapes",
  map.note="Note: Mouseover popup values are
    independent of shape size &amp; color."
)
browseURL("miami_polygons.html")

## End(Not run)
```

ol_geom_circle *OpenLayers Circle Layer*

Description

Function to create a circle layer to add to an OpenLayers Map object.

Usage

```
ol_geom_circle(circle.obj, mapping = ol_aes(), name = NULL,
  df = NULL, toggle.control = FALSE, fill = "#00FF0090",
  fill.opacity = 0.5, lwd = 1, ol.lty = list(), color = "#000000",
  label = NULL, label.params = list(), tooltip = NULL,
  tooltip.params = list())
```

Arguments

`circle.obj` matrix containing three columns: center longitude, center latitude, and radius respectively. Each row yields a single `circle` feature in the resulting layer.

`mapping` list created by ol_aes.

`name` character Layer name.

`df` data.frame with same number of rows as `circle.obj`. Used for aestheic mapping.

`toggle.control` logical indicating whether this layer will have a visibility toggle.

`fill` character color string, or vector of color strings. Used only if no `fill` aesthetic is provided in `mapping`

`fill.opacity` numeric in [0,1]. Controls circle opacity if no opacity provided in `fill` or `fill` aesthetic.

`lwd` numeric circle border width. Used only if no `lwd` aesthetic is provided in `mapping`

`ol.lty` (experimental) numeric vector with length > 1, or `list` of such vectors. Used only if no `ol.lty` aesthetic is provided in `mapping`. See OpenLayers ol/style/Stroke Documentation, 'lineDash' property for more information.

`color` character border color string, or vector of color strings. Used only if no `color` aesthetic is provided in `mapping`

`label` character vector of length `nrow(circle.obj)` of feature labels.

`label.params, tooltip.params`

named lists (e.g., `list(property=value)`) of label and tooltip position and format parameters. See ol_geom_polygon documentation.

`tooltip` character vector of length `nrow(circle.obj)` of feature tooltip popups.

Details

This function stores the data required to generate an OpenLayers vector layer with features using `circle` geometries. See OpenLayers Circle Documentation for details.

Value

A list object of class `Layer.Circle`.

Aesthetics

- `fill`
- `color`
- `lwd`
- `ol_lty` (experimental; See OpenLayers ol/style/Stroke Documentation, 'lineDash' property for more information.)

See Also

`ol_aes`, `ol_map`, `ol_geom_polygon`

Examples

```
miami.circles <- matrix(
    c(
        -80.885+runif(10), #Longitudes
        25.282618+runif(10), #Latitudes
        rnorm(10,2000,500) # Radii in meters
    ),
    ncol=3
)
aesthetic.df <- data.frame(
    type=sample(c("A","B"),10,replace=TRUE),
    value=runif(10)*10
)
miami.OSM.basemap <- ol_map(
    center=c(-80.385790,25.782618),
    zoom=9
    ) +
   streetmap()
circle.layer<-ol_geom_circle(
        miami.circles,
        df = aesthetic.df,
        mapping=ol_aes(fill=type),
        lwd=2,
        name="Meaningless Miami Circles",
        toggle.control=TRUE,
        color="#000000FF",
        tooltip=sprintf("%1.2f",aesthetic.df$value)
        )
circle.fill <- ol_scale_fill_discrete(
        display=TRUE,
        preserve.opacity=TRUE
    )
circles <- miami.OSM.basemap + circle.layer + circle.fill

## Not run:
# Output to file and view
ol_map2HTML(
  circles,
  'miami_circles.html',
  map.heading="Miami Shapes",
  map.note="Note: Mouseover popup values are
    independent of shape size &amp; color."
)
browseURL("miami_circles.html")

## End(Not run)
```

ol_geom_heatmap *OpenLayers Heatmap Layer*

Description

Function to create a Heatmap layer to add to an OpenLayers Map object.

Usage

```
ol_geom_heatmap(point.obj, name = NULL, toggle.control = FALSE,
  gradient = NULL, opacity = 1, radius = 8, blur = 15,
  shadow = 250, weight.values = NULL)
```

Arguments

`point.obj` SpatialPointsDataframe, SpatialPoints, or a matrix containing columns of point longitudes and latitudes, respectively.

`name` character Layer name.

`toggle.control` logical indicating whether this layer will have a visibility toggle.

`gradient` character color gradient of heatmap. See OpenLayers Heatmap Documentation Enclose gradient array in single character string.

`opacity` numeric Heatmap opacity. See OpenLayers Heatmap Documentation.

`radius` numeric Heatmap radius size in pixels.See OpenLayers Heatmap Documentation.

`blur` numeric Heatmap blur. See OpenLayers Heatmap Documentation.

`shadow` numeric Heatmap shadow. See OpenLayers Heatmap Documentation.

`weight.values` numeric vector of weights to be assigned to the points in `point.obj`. Values should be in [0,1].

Details

This function stores the data required to generate an OpenLayers vector layer with features using `Point` geometries. See OpenLayers Heatmap Documentation for details.

Value

A list object of class `Layer.HeatMap`.

Aesthetics

- `fill`
- `size`

See Also

`ol_map`, `ol_geom_polygon`, `ol_geom_circle`, `ol_geom_point`, `ol_geom_icon`

Examples

```
heatmap.pts <- matrix(
    c(
        rnorm(100,-80.385,1), #Miami Longitudes
        rnorm(100,-117.1611,3), #San Diego Longitudes
        rnorm(100,25.782618,1), #Miami Latitudes
        rnorm(100,32.7157,3) # San Diego Latitudes
    ),ncol=2
)
mymap <- ol_map(
    center=c(-98.5,28.5),
    zoom=4
) +
    streetmap() +
    ol_geom_heatmap(
        heatmap.pts,
        name=”Random Heatmap”,
        toggle.control=TRUE,
        opacity=0.25
        )
## Not run:
# Write to file and view in browser
ol_map2HTML(
  mymap,
  "heatmap.html",
  map.note="Heatmap of random points centered on Miami and San Diego."
)
browseURL("heatmap.html")

## End(Not run)
```

ol_geom_icon *OpenLayers Icon Layer*

Description

Function to create a point-icon layer to add to an OpenLayers Map object.

Usage

```
ol_geom_icon(point.obj, src.img = NULL, mapping = ol_aes(),
  name = NULL, df = NULL, toggle.control = FALSE,
  icon.size.scalar = "autoscale", src.img.width = NULL,
  target.icon.width = 30, size.scale.lims = c(0.5, 1.25),
  label = NULL, label.params = list(), tooltip = NULL,
  tooltip.params = list())
```

Arguments

`point.obj` SpatialPointsDataframe, SpatialPoints, or a matrix containing columns of point longitudes and latitudes, respectively.

`src.img` character vector of image file paths.

`mapping` list created by ol_aes. Used for aestheic mapping.

`name` character Layer name.

`df` data.frame with same number of rows as `point.obj` coordinate matrix.

`toggle.control` logical indicating whether this layer will have a visibility toggle.

`icon.size.scalar` numeric scalar vector or ’autoscale’. The width of the icon on the map will be scaled by this input from the original image width. The default ’autoscale’ uses the png, jpeg, or tiff package to scale each image to `target.icon.width`.

`src.img.width` numeric vector of widths of user-supplied images, in pixels. If `icon.size.scalar` is not supplied and `src.img.width` is not provided, image widths will be detected using the png, jpeg, or tiff package.

`target.icon.width` numeric desired width of icons on map, in pixels. Only used if `icon.size.scalar` is ’autoscale’.

`size.scale.lims`
: numeric vector containing the minimum and maximum image scaling for size aesthetic mappings. A value of 1 results renders the image at the default size, determined by `target.icon.width` or `icon.size.scalar`.

`label`
: character vector of point feature labels.

`label.params, tooltip.params`
: named lists (e.g., `list(property=value)`) of label and tooltip position and format parameters. See ol_geom_polygon documentation.

`tooltip`
: character vector of point feature tooltip popups.

Details

This function stores the data required to generate an OpenLayers vector layer with features using `Point` geometries and user-supplied point icons.

Value

A list object of class `Layer.SpatialIcon`.

Aesthetics

- `iconimage`
- `iconsize`

See Also

`ol_map`, `ol_geom_polygon`, `ol_geom_circle`, `ol_geom_line`, `ol_geom_point`

Examples

```
some.r.servers <- matrix(
    c(
        144.964,-37.798,
        -122.920,49.278,
        121.494,31.307,
        25.083,35.307,
        -21.930,64.149,
        11.877,45.407,
        -99.200,19.345,
        5.322,60.388,
        -8.224,39.400,
        -8.616,41.147,
        -73.953,40.768,
        20.304,63.821,
        8.548,47.376,
        33.031,35.247,
        -78.938,36.001,
        -123.279,44.564,
        -96.797,32.777
    ),
    byrow=TRUE,
    ncol=2
)
r.server.names <- c(
    'School of Mathematics and Statistics, University of Melbourne',
    'Simon Fraser University, Burnaby',
    'Shanghai University',
    'University of Crete',
    'Marine Research Institute',
    'University of Padua',
    'Instituto Tecnologico Autonomo de Mexico',
    'University of Bergen',
    'RadicalDevelop, Lda',
    'University of Porto',
    'Four Dots',
    'Academic Computer Club, Umeå University',
    'ETH Zurich',
    'Middle East Technical University Northern Cyprus Campus, Mersin',
    'Duke University, Durham, NC',
    'Oregon State University',
    'Revolution Analytics, Dallas, TX'
)
r.icon <- "https://www.r-project.org/Rlogo.png"
## If width is not provided image must be local
## and png package must be installed.
r.icon.width <- 200
r.map <- ol_map(
    center=c(-100,30),
    zoom=3
) +
    streetmap()+
    ol_geom_icon(
        some.r.servers,
        r.icon,
        name="R Servers",
        icon.size.scalar='autoscale',
        src.img.width=r.icon.width,
        toggle.control=TRUE,
        tooltip=r.server.names
)
## Not run:
# Save as HTML and open in browser
ol_map2HTML(r.map,'R-servers.html')
browseURL("R-servers.html")

## End(Not run)
```

ol_geom_line *OpenLayers Line Layer*

Description

Function to create a line layer to add to an OpenLayers Map object.

Usage

```
ol_geom_line(line.obj, mapping = ol_aes(), name = NULL, df = NULL,
  toggle.control = FALSE, lwd = 1, ol.lty = list(),
  color = "#000000", label = NULL, label.params = list(),
  tooltip = NULL, tooltip.params = list())
```

Arguments

line.obj SpatialLinesDataFrame, SpatialLines, list of lines-like objects, or a two-column matrix of longitude-latitude coordinates to be used as ordered line object coordinates.

mapping list created by ol_aes.

name character Layer name.

df data.frame with same number of lines-like objects as `line.obj`. Used for aestheic mapping. Defaults to line.obj@data if `class(polygon.obj)==SpatialLinesDataFrame)` and df is not provided.

toggle.control logical indicating whether this layer will have a visibility toggle.

lwd numeric line feature width. Used only if no `lwd` aesthetic is provided in `mapping`

ol.lty (experimental) numeric vector with length > 1, or `list` of such vectors. Used only if no `ol.lty` aesthetic is provided in `mapping`. See OpenLayers ol/style/Stroke Documentation, 'lineDash' property for more information.

color character line color string, or vector of color strings. Used only if no `color` aesthetic is provided in `mapping`

label character vector of line feature labels.

label.params, tooltip.params
named lists (e.g., `list(property=value)`) of label and tooltip position and format parameters. See ol_geom_polygon documentation.

tooltip character vector of line feature tooltip popups.

Details

This function creates a list object containing the data required to generate an OpenLayers vector layer with features using `MultiLineString`. See Openlayers MultiLineString Documentation for details.

Value

A list object of class `Layer.SpatialLine`.

Aesthetics

- `color`
- `lwd`
- `ol_lty` (experimental; See OpenLayers ol/style/Stroke Documentation, 'lineDash' property for more information.)

See Also

`ol_aes`, `ol_map`, `ol_geom_point`, `ol_geom_polygon`, `ol_geom_circle`

Examples

```
line.matrix1 <- matrix(
    c(
        -80.4,-80.4,
        25.78,25.88
    ),
    ncol=2
)
line.matrix2 <- matrix(
    c(
        -80.25,-80.35,
        25.65,25.65
    ),
    ncol=2
)
line.list <- list(line.matrix1,line.matrix2)
line.df <- data.frame(
    direction=c("vertical","horizontal"),
    no=c(1,2)
)
miami.gray.basemap <- ol_map(
    center=c(-80.385790,25.782618),
    zoom=9
    ) +
   lightgray()
line.layer <- ol_geom_line(
    line.list,
    mapping=ol_aes(
        color=no,
        lwd=direction
    ),
    df=line.df,
    name="Miami Lines",
    toggle.control=TRUE,
    tooltip=line.df$direction
)
line.color.scale <- ol_scale_color_continuous(
    name="Number",
    display=TRUE
)
line.width.scale <- ol_scale_lwd_discrete(
    lwd.vector=c(
        horizontal=2,
        vertical=4
    ),
    name="Direction",
    display=TRUE
)
line.map.miami <- miami.gray.basemap +
    line.layer +
    line.color.scale +
    line.width.scale
## Not run:
# Output to file and view
ol_map2HTML(
  line.map.miami,
  'miami_lines.html',
  map.heading="Miami Lines"
)
browseURL("miami_lines.html")

## End(Not run)
```

ol_geom_point *OpenLayers Point Layer*

Description

Function to create a points layer to add to an OpenLayers Map object.

Usage

```
ol_geom_point(point.obj, mapping = ol_aes(), name = NULL, df = NULL,
  toggle.control = FALSE, fill = "#00FF00", fill.opacity = 1,
  marker = "pin", size = 0.5, label = NULL, label.params = list(),
  tooltip = NULL, tooltip.params = list())
```

Arguments

`point.obj` SpatialPointsDataframe, SpatialPoints, or a matrix containing columns of point longitudes and latitudes, respectively.

`mapping` list created by ol_aes. Used for aestheic mapping.

`name` character Layer name.

`df` data.frame with same number of rows as `point.obj` coordinate matrix.

`toggle.control` logical indicating whether this layer will have a visibility toggle.

`fill` character color string, or vector of color strings. Used only if no `fill` aesthetic is provided in `mapping`

`fill.opacity` numeric in [0,1]. Controls circle opacity if no opacity provided in `fill` or `fill` aesthetic.

`marker` character. The 'pin' marker draws map pointers similar to most web map applications. The 'dot' or 'point' markers render as cicular points on the map. Other marker types are not supported by this method.

`size` numeric point icon size scalar or vector scalars. Used only if no `size` aesthetic is provided in `mapping`. A value of 1 translates to an icon width of 40 pixels for "pin" markers, or 20 pixels for "dot" markers.

`label` character vector of point feature labels.

`label.params, tooltip.params`
named lists (e.g., `list(property=value)`) of label and tooltip position and format parameters. See ol_geom_polygon documentation.

`tooltip` character vector of point feature tooltip popups.

Details

This function stores the data required to generate an OpenLayers vector layer with features using `Point` geometries. See OpenLayers Point Documentation for details.

Value

A list object of class `Layer.SpatialPoint`.

Aesthetics

- fill
- size

See Also

ol_aes, ol_map, ol_geom_polygon, ol_geom_circle, ol_geom_line, ol_geom_icon

Examples

```
point.matrix <- matrix(
    c(
        -80.885+runif(10),
        25.282618+runif(10)
    ),
    ncol=2
)
point.df <- data.frame(
    pt.type=sample(c("A","B"),10,replace=TRUE),
    pt.value=runif(10)*10
)
miami.map <- ol_map(
    center=c(-80.385790,25.782618),
    zoom=9
) +
    streetmap()
miami.points <- ol_geom_point(
    point.matrix,
    df=point.df,
    mapping=ol_aes(fill=pt.type,size=pt.value),
    name="Random Points of Interest",
    marker="pin",
    toggle.control=TRUE,
    tooltip=point.df$pt.type
)
size.scale <- ol_scale_size_continuous(
    display=TRUE,
    draw.fill='green'
)
fill.scale <- ol_scale_fill_discrete(
    c(B='red',A='green'),
    display=TRUE
)
miami.points.map <- miami.map +
    miami.points +
    size.scale +
    fill.scale

## Not run:
# Output to file and view
ol_map2HTML(
  miami.points.map,
  'Miami_points.html'
)
browseURL('Miami_points.html')
```

```
## End(Not run)
```

`ol_geom_polygon` *OpenLayers Polygon Layer*

Description

Function to create a polygon layer to add to an OpenLayers Map object.

Usage

```
ol_geom_polygon(polygon.obj, mapping = ol_aes(), name = NULL,
  df = NULL, toggle.control = FALSE, fill = "#00FF00",
  fill.opacity = 0.5, lwd = 1, ol.lty = list(), color = "#000000",
  label = NULL, label.params = list(), tooltip = NULL,
  tooltip.params = list())
```

Arguments

`polygon.obj` SpatialPolygonsDataFrame, SpatialPolygons, list of polygon-like objects, or a two-column matrix of longitude-latitude coordinates to be used as ordered polygon vertices.

`mapping` list created by ol_aes.

`name` character Layer name.

`df` data.frame with same number of polygon objects as `polygon.obj`. Used for aestheic mapping. Defaults to polygon.obj@data if `class(polygon.obj)==SpatialPolygonsDataFra` and df is not provided.

`toggle.control` logical indicating whether this layer will have a visibility toggle.

`fill` character color string, or vector of color strings. Used only if no `fill` aesthetic is provided in `mapping`

`fill.opacity` numeric in [0,1]. Controls circle opacity if no opacity provided in `fill` or `fill` aesthetic.

`lwd` numeric polygon border width. Used only if no `lwd` aesthetic is provided in `mapping`

`ol.lty` (experimental) numeric vector with length > 1, or `list` of such vectors. Used only if no `ol.lty` aesthetic is provided in `mapping`. See OpenLayers ol/style/Stroke Documentation, 'lineDash' property for more information.

`color` character border color string, or vector of color strings. Used only if no `color` aesthetic is provided in `mapping`

`label` character vector of polygon feature labels.

`label.params` named list (e.g., `list(property=value)`) of label position and format parameters. See below.

`tooltip` character vector polygon feature tooltip popups.

`tooltip.params` named list (e.g., `list(property=value)`) of tooltip position and format parameters. See below.

Details

This function creates a list object containing the data required to generate an OpenLayers vector layer with features using `MultiPolygon`. See OpenLayers MultiPolygon Documentation for details.

Value

A list object of class `Layer.SpatialPolygon`.

Aesthetics

- `fill`
- `color`
- `lwd`
- `ol_lty` (experimental; See OpenLayers ol/style/Stroke Documentation, 'lineDash' property for more information.)

Formatting Labels With `label.params`

The `label.params` parameter provide direct access to OpenLayers feature text styling (see OpenLayers Documentation). Multiple values for any of these properties is not supported. The following ol/style/Text properties are supported:

`font`	character label font CSS string
`offsetX`	numeric label x-offset
`offsetY`	numeric label y-offset
`rotation`	numeric label rotation
`textAlign`	character label text horizontal alighment
`textBaseline`	character label text vertical alignment
`stroke_color`	character text color
`fill_color`	character text fill color

Formatting Tooltips With `label.params`

The `tooltip.params` parameter enable the user to control tooltip formats. Unlike the `label.params`, not all `tooltip.params` are embedded in Openlayers javascript objects; some are translated to corresponding CSS properties. The table below provides a list of supported properties and their descriptions. OpenLayers Overlay Documentation provides additional information about Overlay Properties.

`font`	character tootltip CSS font
`offsetX`	numeric OpenLayers Overlay x-offset
`offsetY`	numeric OpenLayers Overlay y-offset
`positioning`	character OpenLayers Overlay positioning string
`stroke_color`	character Tooltip CSS font-color
`fill_color` character	Tooltip CSS background-color
`padding`,	character Tooltip CSS padding
`border`,	character Tooltip CSS border
`borderradius` character	Tooltip CSS border-radius

See Also

ol_aes, ol_map, ol_geom_point, ol_geom_line, ol_geom_circle

Examples

```
polygon.matrix1 <- matrix(
    c(
        -80.385+c(0,0.05,0.05,0,0),
        25.782618+c(0,0,0.05,0.05,0)
    ),
    ncol=2
)
polygon.matrix2 <- matrix(
    c(
        -80.34+c(0,0.05,0.025,0),
        25.73++c(0,0,0.025*sqrt(3),0)
    ),
    ncol=2
)
polygon.list<-list(polygon.matrix1,polygon.matrix2)
polygon.df <- data.frame(shape=c("rectangle","triangle"),no=c(1,2))
miami.OSM.basemap <- ol_map(
    center=c(-80.385790,25.782618),
    zoom=9
    ) +
   streetmap()
polygon.layer <- ol_geom_polygon(
    polygon.list,
    mapping=ol_aes(
        fill=shape,
    ),
    df=polygon.df,
    lwd=1,
    name="Miami Polygons",
    toggle.control=TRUE,
    tooltip=polygon.df$no
)
polygon.fill.scale <- ol_scale_fill_discrete(display=TRUE)
polygons.over.miami <- miami.OSM.basemap +
    polygon.layer +
    polygon.fill.scale

## Not run:
# Output to file and view
ol_map2HTML(
  polygons.over.miami,
  'miami_polygons.html',
  map.heading="Miami Shapes",
  map.note="Note: Mouseover popup values are
    independent of shape size &amp; color."
)
browseURL("miami_polygons.html")

## End(Not run)
```

ol_geom_text *OpenLayers Text Layer*

Description

Function to create a Text layer to add to an OpenLayers Map object.

Usage

```
ol_geom_text(point.obj, label, name = NULL, toggle.control = FALSE,
  label.params = list(), tooltip = NULL, tooltip.params = list())
```

Arguments

`point.obj`	SpatialPointsDataframe, SpatialPoints, or a matrix containing columns of point longitudes and latitudes, respectively.
`label`	character vector of text labels to put at points.
`name`	character Layer name.
`toggle.control`	logical indicating whether this layer will have a visibility toggle.
`label.params`	named list (e.g., `list(property=value)`) of label position and format parameters. See below.
`tooltip`	character vector of point feature tooltip popups.
`tooltip.params`	named list (e.g., `list(property=value)`) of tooltip position and format parameters. See ol_geom_polygon documentation.

Details

This function stores the data required to generate an OpenLayers vector layer with text features using `Point` geometries. It does not enable aesthetic mappings to variables.

Value

A list object of class `Layer.Text`.

Formatting Labels With `label.params`

The `label.params` parameter provide direct access to OpenLayers feature text styling (see OpenLayers Documentation). Multiple values for any of these properties is not supported. The following ol/style/Text properties are supported:

`font`	character label font CSS string
`offsetX`	numeric label x-offset
`offsetY`	numeric label y-offset
`rotation`	numeric label rotation
`textAlign`	character label text horizontal alighment
`textBaseline`	character label text vertical alignment
`stroke_color`	character text color
`fill_color`	character text fill color

See Also

ol_map, ol_geom_point

Examples

```
text.pts <- matrix(
    c(
        -101.5, 39.2,
        -101.1, 54,
        -101.1, 21.4
    ),
    byrow=TRUE,
    ncol=2
)
text.labels <- c("USA","Canada","Mexico")
mymap <- ol_map(
    center=c(-100,25),
    zoom=3
) +
    oceanbase() +
    ol_geom_text(
        text.pts,
        text.labels,
        toggle.control=TRUE,
        label.params=list(
            font="16px sans-serif",
            stroke_color="#FF0000",
            fill_color="#FFFFFF00"
        )
    )
## Not run:
# Write to file and view in browser
ol_map2HTML(mymap, "textmap.html")
browseURL("textmap.html")

## End(Not run)
```

ol_map *OpenLayers Map*

Description

Create an OpenLayers Map Object.

Usage

```
ol_map(zoom = 10, center = c(-117.1611, 32.7157))
```

Arguments

zoom integer map initial zoom level.

center numeric vector of length 2 containing decimal longitude and latitude coordinates for initial map center.

Details

This function creates a new S3 OpenLayers Map object with no layers. If `ol.source.url` is `NULL` and `nga.olsource` is `FALSE`, OpenLayers Javascript source will be embedded directly into the HTML when `ol_map2HTML` or `ol_map2Strings` is called. Otherwise, the output HTML/Javascript with source the OpenLayers library according to the value of `ol.source.url`, or the NGA hosted OpenLayers library if `nga.olsource` is `TRUE`.

Value

A list object of class `Ol.Map`.

See Also

`ol_map2HTML`, `ol_map2Strings`, `nga_basemap`, `user_arcgis_basemap`

Examples

```
miami.OSM.basemap <- ol_map(
    center=c(-80.385790,25.782618),
    zoom=9
    ) +
   streetmap()
## Not run:
ol_map2HTML(
  miami.OSM.basemap,
  'miami.html',
  map.heading="Miami, FL"
)
browseURL("miami.html")

## End(Not run)
```

`ol_map2HTML` *Export OpenLayers Map to file.*

Description

Writes Ol.Map object to HTML file.

Usage

```
ol_map2HTML(ol.map.obj, file.name, page.name = "ROpenLayers Map",
  width = NULL, height = NULL, ol.source.url = NULL,
  nga.olsource = FALSE, map.heading = NULL, map.note = NULL,
  nice.format = FALSE, IE.compatability.view = TRUE)
```

Arguments

`ol.map.obj`	Ol.Map object to be exported.
`file.name`	character output HTML file name.
`page.name`	character page title to be included in the HTML head section.

`width`	numeric or character CSS value width of map container.
`height`	numeric or character CSS value height of map container.
`ol.source.url`	character string containing the url to the OpenLayers javascript library. Ignored if nga.olsource is `TRUE`.
`nga.olsource`	logical. `TRUE` will use the OpenLayers 3.16.0 javascript library from `http://home.gvs.nga.smil.mil/` (requires authentication); `FALSE` uses the sources the `ol.source.url`, if provided, or embeds the OpenLayers 5.3.0 JavaScript code in the HTML head. Only used if `ol.source.url` is missing or `NULL`.
`map.heading`	character heading to be placed over map in html h1 tag.
`map.note`	character note placed in html paragraph (<p>) tag centered under map container.
`nice.format`	logical. If `TRUE`, output file will be formated with new lines and indentation for human readability.
`IE.compatability.view`	logical. If `TRUE`, the statement `<meta http-equiv="X-UA-Compatible" content="IE=edge"/>` to the HTML document head. This statement is required for some browsers to render the map.

Details

Ol.Map object is written to HTML file with embedded javascript. The file will include or source the OpenLayers javascript library as specified in the Ol.Map object (see `ol_map`). The Javascript will call any REST APIs required for each layer in order to produce an output file with supporting images, if required, that can be placed directly into a directory hosted by a minimal http server.

See Also

`ol_map`, `ol_map2Strings`,

Examples

```
mymap <- ol_map()
base.layer <- lightgray()
mymap <- mymap + base.layer
## Not run:
# The following writes HTML and needed images
ol_map2HTML(mymap,"SanDiego.html", nice.format=TRUE)
# Open in browser
browseURL("SanDiego.html")

## End(Not run)
```

`ol_map2Strings` *OpenLayers Map HTML to List*

Description

Assigns Ol.Map HTML content to list.

Usage

```
ol_map2Strings(ol.map.obj, width = NULL, height = NULL,
  ol.source.url = NULL, nga.olsource = FALSE, map.heading = NULL,
  map.note = NULL)
```

Arguments

`ol.map.obj` Ol.Map object to be exported.

`width` numeric or character CSS value width of map container.

`height` numeric or character CSS value height of map container.

`ol.source.url` character string containing the url to the OpenLayers javascript library. Ignored if nga.olsource is `TRUE`.

`nga.olsource` logical. `TRUE` will use the OpenLayers 3.16.0 javascript library from `https://home.gvs.nga.mil` (requires authentication); `FALSE` uses the sources the `ol.source.url`, if provided, or embeds the OpenLayers 3.21.1 JavaScript code in the HTML head. Only used if `ol.source.url` is missing or `NULL`.

`map.heading` character heading to be placed over map in html h1 tag.

`map.note` character note placed in html paragraph (<p>) tag centered under map container.

Details

Ol.Map object HTML is exported to a list object that can be deployed in a variety of applications or server environments. See exmaples for a minimal example using RShiny. This method does not currently support adding multiple maps to the same web page, as javascript variable names would be replicated.

Value

list object with the following character elements:

`$head.meta.IE.compatibility` HTML meta tag for IE compatability viewing.

`$head.script` HTML script block including or sourcing the OpenLayers Javascript library (see `ol.m`

`$style` CSS code for styling the map and legends.

`$body.html` HTML map and legend containers, and associated elements.

`$body.script` Javascript code writing the layer and map objects.

See Also

`ol_map`, `ol_map2HTML`,

Examples

```
heatmap.pts <- matrix(
    c(
        rnorm(100,-80.385,1), #Miami Longitudes
        rnorm(100,-117.1611,3), #San Diego Longitudes
        rnorm(100,25.782618,1), #Miami Latitudes
        rnorm(100,32.7157,3) # San Diego Latitudes
    ),ncol=2
)
mymap <- ol_map(
    center=c(-98.5,28.5),
    zoom=4
 ) +
    streetmap() +
    ol_geom_heatmap(
        heatmap.pts,
        name="Random Heatmap",
        toggle.control=TRUE,
        opacity=0.25
        )
## The following line will create image files
## as needed for point layers and legends.
## None are required in this example.
HTML.strings <- ol_map2Strings(
  mymap,
  nga.olsource=FALSE,
  map.note="Heatmap of random points centered on Miami and San Diego."
)
## Minimal shiny example
## Not run:
library(shiny)
ui <- shinyUI(
    fluidPage(
        #Add OpenLayers Javascript source & CSS to head
        tags$head(
            HTML(HTML.strings[[1]])
            HTML(HTML.strings[[2]]),
            tags$style(HTML(HTML.strings[[3]]))
        ),
        titlePanel("Random Heatmap"),
        mainPanel(
            tags$div(HTML(HTML.strings[[4]]))
        ),
        tags$script(HTML(HTML.strings[[5]]))
    )
)
server <- function(input,output){
}
shinyApp(ui=ui,server)

## End(Not run)
```

ol_scale_color_continuous *Line Color Scale (Continuous)*

Description

Specify a line (or border) color mapping scale.

Usage

```
ol_scale_color_continuous(low.val, high.val, low.col = NULL,
  high.col = NULL, rotate.clockwise = TRUE, name = NULL,
  na.col.val = "#FFFFFF00", opacity = 1, preserve.opacity = NULL,
  display = FALSE)
```

Arguments

`low.val` numeric the minimum variable value to be mapped to the lowest color.

`high.val` numeric the maximum variable value to be mapped to the highest color.

`low.col` character the "low" color.

`high.col` character the "high" color.

`rotate.clockwise` logical. If `TRUE`, continuous scale will map to colors on a clockwise rotation from `low.col` to `high.col` on the hue-saturation-value (HSV) color space. If `FALSE`, rotation will be counter-clockwise.

`name` character the scale name.

`na.col.val` character the color assigned to non-numeric or NA values.

`opacity` numeric in [0,1]. The fill opacity, if not specified in the `low.col` and `high.col` colors.

`preserve.opacity` logical indicating whether to draw the legend with the same opacity as the feature fills on the map.

`display` logical indicating whether to draw the scale for output in the HTML. If `TRUE`, a bitmap will be created and sourced in the HTML in an `img` tag.

Details

This method maps OpenLayers feature line or border colors to continuous variable values. This scale can be added to an Ol.Map S3 object only if the Ol.Map object has a layer with a "color" mapping to a numeric variable. If no such layer exists, attempts to add this type of scale will result in a warning. Attempts to apply this scale to a non-numeric variable will throw an error.

Value

list of class `Scale.Color.Continuous`.

See Also

`ol_map`, `ol_geom_polygon`, `ol_geom_line`

Examples

```
line.matrix1 <- matrix(
    c(
        -80.4,-80.4,
        25.78,25.88
    ),
    ncol=2
)
line.matrix2 <- matrix(
    c(
        -80.25,-80.35,
        25.65,25.65
    ),
    ncol=2
)
line.list <- list(line.matrix1,line.matrix2)
line.df <- data.frame(
    direction=c("vertical","horizontal"),
    no=runif(2)
)
miami.gray.basemap <- ol_map(
    center=c(-80.385790,25.782618),
    zoom=9
    ) +
   lightgray()
line.layer <- ol_geom_line(
    line.list,
    mapping=ol_aes(
        color=no #continuous mapping
    ),
    df=line.df,
    name="Miami Lines",
    toggle.control=TRUE,
    lwd=5,
    tooltip=line.df$no
)
line.color.scale <- ol_scale_color_continuous(
    low.val = 0,
    high.val = 1,
    low.col = 'red',
    high.col= 'green',
    opacity = 1,
    preserve.opacity = TRUE,
    name = "Number",
    display = TRUE
)
line.map.miami <- miami.gray.basemap +
    line.layer +
    line.color.scale

## Not run:
# Output to file and view
ol_map2HTML(
  line.map.miami,
  'miami_lines.html',
  map.heading="Miami Lines"
)
browseURL("miami_lines.html")

## End(Not run)
```

ol_scale_color_discrete *Line Color Scale (Discrete)*

Description

Specify a discrete line color mapping scale.

Usage

```
ol_scale_color_discrete(color.vector = NULL, name = NULL,
  na.col.val = "#FFFFFF00", ordered.values = NULL, opacity = 1,
  preserve.opacity = FALSE, draw.lty = "solid", draw.lwd = 3,
  display = FALSE)
```

Arguments

`color.vector`	character named vector of the form `c(value=color)`. If `NULL`, a default color mapping is assigned.
`name`	character the scale name.
`na.col.val`	character the color assigned to unrecognized or NA values.
`ordered.values`	character, numeric, or factor vector containing the ordered unique discrete variable values. This input is used to determine the order of the values appearing in the legend. If not supplied, the order is taken from `names(color.vector)`.
`opacity`	numeric in [0,1]. The color opacity, if not specified in the `color.vector` colors.
`preserve.opacity`	logical indicating whether to draw the legend with the same opacity as the feature fills on the map.
`draw.lty`	character indicating the line type for the legend only. This will be passed to an R plot command. See `par`.
`draw.lwd`	numeric width of lines only used in drawing the legend.
`display`	logical indicating whether to draw the scale for output in the HTML. If `TRUE`, a bitmap will be created and sourced in the HTML in an `img` tag. If `FALSE`, the `draw.*` inputs are ignored.

Details

This method maps OpenLayers feature line colors to discrete variable values. This scale can be added to an Ol.Map S3 object only if the Ol.Map object has a layer with a "color" mapping to a numeric, character, or factor type variable. If no such layer exists, attempts to add this type of scale will result in a warning. The `color.vector` input enables the user to specify the exact mapping, assigning colors to specific variable values.

Value

list of class `Scale.Color.Discrete`.

See Also

`ol_map`, `ol_geom_polygon`, `ol_geom_line`

Examples

```
line.matrix1 <- matrix(
    c(
        -80.4,-80.4,
        25.78,25.88
    ),
    ncol=2
)
```

```
line.matrix2 <- matrix(
    c(
        -80.25,-80.35,
        25.65,25.65
    ),
    ncol=2
)
line.list <- list(line.matrix1,line.matrix2)
line.df <- data.frame(
    direction=c("vertical","horizontal"),
    no=runif(2)
)
miami.gray.basemap <- ol_map(
    center=c(-80.385790,25.782618),
    zoom=9
    ) +
   lightgray()
line.layer <- ol_geom_line(
    line.list,
    mapping=ol_aes(
        color=direction # discrete mapping
    ),
    df=line.df,
    name="Miami Lines",
    toggle.control=TRUE,
    lwd=5
)
line.color.scale <- ol_scale_color_discrete(
    color.vector=c(
        vertical = 'red',
        horizontal = 'blue'
    ),
    name="Direction",
    display=TRUE
)
line.map.miami <- miami.gray.basemap +
    line.layer +
    line.color.scale
## Not run:
# Output to file and view
ol_map2HTML(
  line.map.miami,
  'miami_lines.html',
  map.heading="Miami Lines"
)
browseURL("miami_lines.html")

## End(Not run)
```

ol_scale_fill_continuous

Fill Color Scale (Continuous)

Description

Specify a continuous fill color mapping scale.

Usage

```
ol_scale_fill_continuous(low.val, high.val, low.col = NULL,
  high.col = NULL, rotate.clockwise = TRUE, name = NULL,
  na.col.val = "#FFFFFF00", opacity = 1, preserve.opacity = FALSE,
  display = FALSE)
```

Arguments

`low.val` numeric the minimum variable value to be mapped to the lowest color.

`high.val` numeric the maximum variable value to be mapped to the highest color.

`low.col` character the "low" color.

`high.col` character the "high" color.

`rotate.clockwise` logical. If `TRUE`, continuous scale will map to colors on a clockwise rotation from `low.col` to `high.col` on the hue-saturation-value (HSV) color space. If `FALSE`, rotation will be counter-clockwise.

`name` character the scale name.

`na.col.val` character the color assigned to non-numeric or NA values.

`opacity` numeric in [0,1]. The fill opacity, if not specified in the `low.col` and `high.col` colors.

`preserve.opacity` logical indicating whether to draw the legend with the same opacity as the feature fills on the map.

`display` logical indicating whether to draw the scale for output in the HTML. If `TRUE`, a bitmap will be created and sourced in the HTML in an `img` tag.

Details

This method maps OpenLayers feature fill colors to continuous variable values. This scale can be added to an Ol.Map S3 object only if the Ol.Map object has a layer with a "fill" mapping to a numeric variable. If no such layer exists, attempts to add this type of scale will result in a warning. Attempts to add this scale to a discrete variable mapping will throw an error.

Value

list of class `Scale.Fill.Continuous`.

See Also

`ol_map`, `ol_geom_polygon`, `ol_geom_circle`

Examples

```
polygon.matrix1 <- matrix(
    c(
        -80.385+c(0,0.05,0.05,0,0),
        25.782618+c(0,0,0.05,0.05,0)
    ),
    ncol=2
)
polygon.matrix2 <- matrix(
    c(
        -80.34+c(0,0.05,0.025,0),
        25.73++c(0,0,0.025*sqrt(3),0)
    ),
    ncol=2
)
polygon.list<-list(polygon.matrix1,polygon.matrix2)
polygon.df <- data.frame(shape=c("rectangle","triangle"),no=runif(2))
miami.OSM.basemap <- ol_map(
    center=c(-80.385790,25.782618),
    zoom=9
    ) +
   streetmap()
polygon.layer <- ol_geom_polygon(
    polygon.list,
    mapping=ol_aes(
        fill=no #numeric mapping
    ),
    df=polygon.df,
    lwd=1,
    name="Miami Polygons",
    toggle.control=TRUE,
    tooltip=polygon.df$no
)
polygon.fill.scale <- ol_scale_fill_continuous(
    low.val=0,
    high.val=1,
    low.col='red',
    high.col='green',
    opacity=0.5,
    preserve.opacity=FALSE,
    display=TRUE
)
polygons.over.miami <- miami.OSM.basemap +
    polygon.layer +
    polygon.fill.scale

## Not run:
# Output to file and view
ol_map2HTML(
  polygons.over.miami,
  'miami_polygons.html',
  map.heading="Miami Shapes",
  map.note="Note: Mouseover popup values are
    independent of shape size &amp; color."
)
browseURL("miami_polygons.html")

## End(Not run)
```

ol_scale_fill_discrete *Fill Color Scale (Discrete)*

Description

Specify a discrete fill color mapping scale.

Usage

```
ol_scale_fill_discrete(color.vector = NULL, name = NULL,
  na.col.val = "#FFFFFF00", ordered.values = NULL,
  ordinal.scale = FALSE, opacity = 1, preserve.opacity = FALSE,
  draw.lines = NULL, draw.color = "black", draw.lty = "solid",
  draw.lwd = 1, display = FALSE)
```

Arguments

`color.vector`	character named vector of the form `c(value=color)`. If `NULL`, a default color mapping is assigned.
`name`	character the scale name.
`na.col.val`	character the color assigned to unrecognized or NA values.
`ordered.values`	character, numeric, or factor vector containing the ordered unique discrete variable values. This input is used to determine the order of the values appearing in the legend. If not supplied, the order is taken from `names(color.vector)`.
`ordinal.scale`	logical. If `TRUE`, the colors in the legend will not have spaces between them.
`opacity`	numeric in [0,1]. The fill opacity, if not specified in the `color.vector` colors.
`preserve.opacity`	logical indicating whether to draw the legend with the same opacity as the feature fills on the map.
`draw.lines`	logical indicating whether to draw a border around each color in the legend. If `NULL`, a default is assigned according to the type of layer containing the scale.
`draw.color`	character color of the border in the legend.
`draw.lty`	character indicating the border line type for the legend only. This will be passed to an R plot command. See `par`.
`draw.lwd`	numeric width of border only used in drawing the legend.
`display`	logical indicating whether to draw the scale for output in the HTML. If `TRUE`, a bitmap will be created and sourced in the HTML in an `img` tag. If `FALSE`, the `draw.*` inputs are ignored.

Details

This method maps OpenLayers feature fill colors to discrete variable values. This scale can be added to an Ol.Map S3 object only if the Ol.Map object has a layer with a "fill" mapping to a numeric, character, or factor type variable. If no such layer exists, attempts to add this type of scale will result in a warning. The `color.vector` input enables the user to specify the exact mapping, assigning colors to specific variable values.

Value

list of class `Scale.Fill.Discrete`.

See Also

`ol_map`, `ol_geom_polygon`, `ol_geom_circle`

Examples

```
polygon.matrix1 <- matrix(
    c(
        -80.385+c(0,0.05,0.05,0,0),
        25.782618+c(0,0,0.05,0.05,0)
    ),
    ncol=2
)
polygon.matrix2 <- matrix(
    c(
        -80.34+c(0,0.05,0.025,0),
        25.73++c(0,0,0.025*sqrt(3),0)
    ),
    ncol=2
)
polygon.list<-list(polygon.matrix1,polygon.matrix2)
polygon.df <- data.frame(shape=c("rectangle","triangle"),no=runif(2))
miami.OSM.basemap <- ol_map(
    center=c(-80.385790,25.782618),
    zoom=9
    ) +
   streetmap()
polygon.layer <- ol_geom_polygon(
    polygon.list,
    mapping=ol_aes(
        fill=shape #discrete mapping
    ),
    df=polygon.df,
    lwd=1,
    name="Miami Polygons",
    toggle.control=TRUE
)
polygon.fill.scale <- ol_scale_fill_discrete(
    color.vector=c(
        rectangle = 'red',
        triangle = 'blue'
    ),
    name = "Shape",
    opacity = 0.5,
    preserve.opacity = FALSE,
    display = TRUE
)
polygons.over.miami <- miami.OSM.basemap +
    polygon.layer +
    polygon.fill.scale

## Not run:
# Output to file and view
ol_map2HTML(
  polygons.over.miami,
  'miami_polygons.html',
  map.heading="Miami Shapes",
  map.note="Note: Mouseover popup values are
    independent of shape size &amp; color."
)
browseURL("miami_polygons.html")
```

```
## End(Not run)
```

ol_scale_iconimage_discrete
Icon Image Scale

Description

Map icon images to discrete variable values.

Usage

```
ol_scale_iconimage_discrete(icon.img.vector = NULL, name = NULL,
  na.img.src = NULL, ordered.values = NULL, display = FALSE,
  icon.width = NULL)
```

Arguments

`icon.img.vector`	character named vector of the form `c(value=image.path)`. If `NULL`, a default mapping is assigned to the images available in the layer.
`name`	character the scale name.
`na.img.src`	character image path assigned to unrecognized or NA values.
`ordered.values`	character, numeric, or factor vector giving the ordering for the legend display. If `NULL`, the ordering from `icon.img.vector` is used.
`display`	logical indicating whether to draw the scale for output in the HTML. If `TRUE`, a bitmap will be created and sourced in the HTML in one or more `img` tags.
`icon.width`	numeric width(s) of icons for the legend display only. Icon widths for the map overlay are defined in `ol_geom_icon`.

Details

This method maps OpenLayers point (icon) feature images to discrete variable values. This scale can be added to an Ol.Map S3 object only if the Ol.Map object has a layer with an "iconimage" mapping to a numeric, character, or factor variable. If no such layer exists, attempts to add this type of scale will result in a warning.

Value

list of class `Scale.IconImage.Discrete`.

See Also

`ol_map`, `ol_geom_icon`,

Examples

```
freebsd.icon <- "https://www.freebsd.org/gifs/daemon-phk.png"
freebsd.icon.width <- 191
r.icon <- "https://www.r-project.org/Rlogo.png"
r.icon.width <- 200
loc.df <- data.frame(
    lon=c(
        -73.953,
        -78.938,
        -74.007
    ),
    lat=c(
        40.768,
        36.001,
        40.708
    ),
    type=c(
        "R",
        "R",
        "BSD"
    )

)
icon.map <- ol_map(
    center=c(-75,38),
    zoom=5
) +
    streetmap()+
    ol_geom_icon(
        loc.df[,1:2],
        c(r.icon,freebsd.icon),
        mapping=ol_aes(iconimage=type),
        df = loc.df,
        name="Some Open Source Locations",
        icon.size.scalar='autoscale',
        src.img.width=c(r.icon.width,freebsd.icon.width),
        toggle.control=TRUE
) +
    ol_scale_iconimage_discrete(
    c(R=r.icon,BSD=freebsd.icon),
    display=TRUE
)
## Not run:
# Oave as HTML and open in browser
ol_map2HTML(
  icon.map,
  'servers.html'
)
browseURL("servers.html")

## End(Not run)
```

---

ol_scale_iconsize_continuous

Icon Size Scale (Continuous)

Description

Specify a continuous size mapping for an icon layer.

Usage

```
ol_scale_iconsize_continuous(low.val, high.val, low.size = 0.33,
  high.size = 0.75, name = NULL, na.size.val = 0.33,
  legend.breaks = NULL, display = FALSE, display.icon.img.src = NULL)
```

Arguments

`low.val` numeric the minimum variable value to be mapped to the smallest size.

`high.val` numeric the maximum variable value to be mapped to the largest size.

`low.size` numeric smallest size scalar.

`high.size` numeric largest size scalar.

`name` character the scale name.

`na.size.val` numeric the size scalar assigned to non-numeric or NA values.

`legend.breaks` numeric ordered vector of variable values to display in the legend.

`display` logical indicating whether to draw the scale for output in the HTML. If `TRUE`, a bitmap will be created and sourced in the HTML in one or more `img` tags.

`display.icon.img.src`
character path to image file to use in size legend. If NULL, the first image supplied to the icon layer will be used.

Details

This method maps OpenLayers point (icon) feature sizes to continuous variable values. This scale can be added to an Ol.Map S3 object only if the Ol.Map object has a layer with an "iconsize" mapping to a numeric variable. If no such layer exists, attempts to add this type of scale will result in a warning. Attempt to apply this scale to a non-numeric variable will throw an error.

Size inputs to this method are applied as scalars to the icon widths. A size value of 1 translates to the `target.icon.width` assigned to the layer.

Value

list of class `Scale.IconSize.Continuous`.

See Also

`ol_map`, `ol_geom_icon`,

Examples

```
some.r.servers <- matrix(
    c(
        144.964,-37.798,
        -122.920,49.278,
        121.494,31.307,
        25.083,35.307,
        -21.930,64.149,
        11.877,45.407,
        -99.200,19.345,
        5.322,60.388,
        -8.224,39.400,
        -8.616,41.147,
        -73.953,40.768,
        20.304,63.821,
        8.548,47.376,
        33.031,35.247,
        -78.938,36.001,
        -123.279,44.564,
        -96.797,32.777
    ),
    byrow=TRUE,
    ncol=2
)
r.server.df <- data.frame(
    server.name=c(
        'School of Mathematics and Statistics, University of Melbourne',
        'Simon Fraser University, Burnaby',
        'Shanghai University',
        'University of Crete',
        'Marine Research Institute',
        'University of Padua',
        'Instituto Tecnologico Autonomo de Mexico',
        'University of Bergen',
        'RadicalDevelop, Lda',
        'University of Porto',
        'Four Dots',
        'Academic Computer Club, Umeå University',
        'ETH Zurich',
        'Middle East Technical University Northern Cyprus Campus, Mersin',
        'Duke University, Durham, NC',
        'Oregon State University',
        'Revolution Analytics, Dallas, TX'
    ),
    server.value = runif(17)*10,
    stringsAsFactors=FALSE
)
r.icon <- "https://www.r-project.org/Rlogo.png"
## If width is not provided image must be local
## and png package must be installed.
r.icon.width <- 200
r.map <- ol_map(
    center=c(-100,30),
    zoom=3
) +
    streetmap()+
    ol_geom_icon(
        some.r.servers,
        r.icon,
        mapping=ol_aes(iconsize=server.value),
        df = r.server.df,
        name="R Servers",
        icon.size.scalar='autoscale',
        src.img.width=r.icon.width,
        toggle.control=TRUE,
        tooltip=r.server.df$server.name
```

```
) +
    ol_scale_iconsize_continuous(
    low.val=0,
    high.val=10,
    legend.breaks=c(0,2.5,5,7.5,10),
    display=TRUE
)
## Not run:
# Oave as HTML and open in browser
ol_map2HTML(
  r.map,
  'R-servers.html'
)
browseURL("R-servers.html")

## End(Not run)
```

ol_scale_iconsize_discrete

Icon Size Scale (Discrete)

Description

Specify a discrete size mapping for an icon layer.

Usage

```
ol_scale_iconsize_discrete(size.vector = NULL, name = NULL,
  na.size.val = 0.33, legend.breaks = NULL, display = FALSE,
  display.icon.img.src = NULL)
```

Arguments

`size.vector`	numeric named vector of the form `c(value=width)`. If `NULL`, a default size mapping is assigned.
`name`	character the scale name.
`na.size.val`	numeric the size scalar assigned to non-numeric or NA values.
`legend.breaks`	numeric ordered vector of variable values to display in the legend.
`display`	logical indicating whether to draw the scale for output in the HTML. If `TRUE`, a bitmap will be created and sourced in the HTML in one or more `img` tags.
`display.icon.img.src`	character path to image file to use in size legend. If NULL, the first image supplied to the icon layer will be used.

Details

This method maps OpenLayers point (icon) feature sizes to discrete variable values. This scale can be added to an Ol.Map S3 object only if the Ol.Map object has a layer with a "iconsize" mapping to a numeric, character, or factor variable. If no such layer exists, attempts to add this type of scale will result in a warning.

Size inputs to this method are applied as scalars to the icon widths. A size value of 1 translates to the `target.icon.width` assigned to the layer.

Value

list of class Scale.IconSize.Discrete.

See Also

ol_map, ol_geom_icon,

Examples

```
some.r.servers <- matrix(
    c(
        144.964,-37.798,
        -122.920,49.278,
        121.494,31.307,
        25.083,35.307,
        -21.930,64.149,
        11.877,45.407,
        -99.200,19.345,
        5.322,60.388,
        -8.224,39.400,
        -8.616,41.147,
        -73.953,40.768,
        20.304,63.821,
        8.548,47.376,
        33.031,35.247,
        -78.938,36.001,
        -123.279,44.564,
        -96.797,32.777
    ),
    byrow=TRUE,
    ncol=2
)
r.server.df <- data.frame(
    server.name=c(
        'School of Mathematics and Statistics, University of Melbourne',
        'Simon Fraser University, Burnaby',
        'Shanghai University',
        'University of Crete',
        'Marine Research Institute',
        'University of Padua',
        'Instituto Tecnologico Autonomo de Mexico',
        'University of Bergen',
        'RadicalDevelop, Lda',
        'University of Porto',
        'Four Dots',
        'Academic Computer Club, Umeå University',
        'ETH Zurich',
        'Middle East Technical University Northern Cyprus Campus, Mersin',
        'Duke University, Durham, NC',
        'Oregon State University',
        'Revolution Analytics, Dallas, TX'
    ),
    server.type = sample(c("A","B","C"),17,replace=TRUE),
    stringsAsFactors=FALSE
)
r.icon <- "https://www.r-project.org/Rlogo.png"
```

```
## If width is not provided image must be local
## and png package must be installed.
r.icon.width <- 200
r.map <- ol_map(
    center=c(-100,30),
    zoom=3
) +
    streetmap()+
    ol_geom_icon(
        some.r.servers,
        r.icon,
        mapping=ol_aes(iconsize=server.type),
        df = r.server.df,
        name="R Servers",
        icon.size.scalar='autoscale',
        src.img.width=r.icon.width,
        toggle.control=TRUE,
        tooltip=r.server.df$server.type
) +
    ol_scale_iconsize_discrete(
    display=TRUE
)
## Not run:
# Oave as HTML and open in browser
ol_map2HTML(
  r.map,
  'R-servers.html'
)
browseURL("R-servers.html")

## End(Not run)
```

`ol_scale_lty_discrete` *Line Type Scale (Experimental)*

Description

Specify a discrete line type mapping scale.

Usage

```
ol_scale_lty_discrete(lty.list = NULL, name = NULL, na.lty.val = NA,
  ordered.values = NULL, opacity = 1, draw.color = "black",
  draw.lwd = 1, display = FALSE)
```

Arguments

`lty.list`	numeric named list of the form `list(value=numeric.vector)`. If `NULL`, a default line type mapping is assigned. The input vectors will be converted to javascript numeric arrays. See the `lineDash` propery in the OpenLayers Stroke Documentation for interpretation of these inputs.
`name`	character the scale name.
`na.lty.val`	numeric the `lineDash` vector assigned to unrecognized or NA values.

`ordered.values`	character, numeric, or factor vector containing the ordered unique discrete variable values. This input is used to determine the order of the values appearing in the legend. If not supplied, the order is taken from `names(lwd.vector)`.
`opacity`	numeric in [0,1]. The line opacity used in the legend only.
`draw.color`	character color used in the legend lines only.
`draw.lwd`	numeric indicating the line width for the legend only. This will be passed to an R plot command. See `par`.
`display`	logical indicating whether to draw the scale for output in the HTML. If `TRUE`, a bitmap will be created and sourced in the HTML in an `img` tag. If `FALSE`, the `draw.*` inputs are ignored.

Details

This method maps OpenLayers feature line dash types to discrete variable values. This scale can be added to an Ol.Map S3 object only if the Ol.Map object has a layer with a "lty" mapping to a numeric, character, or factor type variable. If no such layer exists, attempts to add this type of scale will result in a warning. The `lty.list` input enables the user to specify the exact mapping, assigning line dash types to specific variable values. The `lty` aesthetic does not have a continuous scale.

Note: this method does not result in consistant rendering for different browsers or map zoom levels.

Value

list of class `Scale.Lwd.Discrete`.

See Also

`ol_map`, `ol_geom_polygon`, `ol_geom_line`

Examples

```
line.matrix1 <- matrix(
    c(
        -80.4,-80.4,
        25.78,25.88
    ),
    ncol=2
)
line.matrix2 <- matrix(
    c(
        -80.25,-80.35,
        25.65,25.65
    ),
    ncol=2
)
line.list <- list(line.matrix1,line.matrix2)
line.df <- data.frame(
    direction=c("vertical","horizontal"),
    no=runif(2)
)
miami.gray.basemap <- ol_map(
    center=c(-80.385790,25.782618),
    zoom=9
    ) +
    lightgray()
line.layer <- ol_geom_line(
    line.list,
    mapping=ol_aes(
        lty=direction # discrete mapping
    ),
    df=line.df,
    name="Miami Lines",
    toggle.control=TRUE
)
line.type.scale <- ol_scale_lty_discrete(
    lty.list=list(
        vertical = 1,
        horizontal = c(5,5)
    ),
    name="Direction",
    display=TRUE
)
line.map.miami <- miami.gray.basemap +
    line.layer +
    line.type.scale
## Not run:
# Output to file and view
ol_map2HTML(
  line.map.miami,
  'miami_lines.html',
  map.heading="Miami Lines"
)
browseURL("miami_lines.html")

## End(Not run)
```

ol_scale_lwd_discrete *Line Width Scale*

Description

Specify a discrete line width mapping scale.

Usage

```
ol_scale_lwd_discrete(lwd.vector = NULL, name = NULL, na.lwd.val = 1,
  ordered.values = NULL, opacity = 1, draw.color = "black",
  draw.lty = "solid", display = FALSE)
```

Arguments

lwd.vector	numeric named vector of the form c(value=width). If NULL, a default width mapping is assigned.
name	character the scale name.
na.lwd.val	numeric the width assigned to unrecognized or NA values.
ordered.values	character, numeric, or factor vector containing the ordered unique discrete variable values. This input is used to determine the order of the values appearing in the legend. If not supplied, the order is taken from names(lwd.vector).

`opacity` numeric in [0,1]. The line opacity used in the legend only.

`draw.color` character color used in the legend lines only.

`draw.lty` character indicating the line type for the legend only. This will be passed to an R plot command. See `par`.

`display` logical indicating whether to draw the scale for output in the HTML. If `TRUE`, a bitmap will be created and sourced in the HTML in an `img` tag. If `FALSE`, the `draw.*` inputs are ignored.

Details

This method maps OpenLayers feature line widths to discrete variable values. This scale can be added to an Ol.Map S3 object only if the Ol.Map object has a layer with a "lwd" mapping to a numeric, character, or factor type variable. If no such layer exists, attempts to add this type of scale will result in a warning. The `lwd.vector` input enables the user to specify the exact mapping, assigning widths to specific variable values. The width aesthetic does not have a continuous scale.

Value

list of class `Scale.Lwd.Discrete`.

See Also

`ol_map`, `ol_geom_polygon`, `ol_geom_line`

Examples

```
line.matrix1 <- matrix(
    c(
        -80.4,-80.4,
        25.78,25.88
    ),
    ncol=2
)
line.matrix2 <- matrix(
    c(
        -80.25,-80.35,
        25.65,25.65
    ),
    ncol=2
)
line.list <- list(line.matrix1,line.matrix2)
line.df <- data.frame(
    direction=c("vertical","horizontal"),
    no=runif(2)
)
miami.gray.basemap <- ol_map(
    center=c(-80.385790,25.782618),
    zoom=9
    ) +
   lightgray()
line.layer <- ol_geom_line(
    line.list,
    mapping=ol_aes(
        lwd=direction # discrete mapping
    ),
    df=line.df,
    name="Miami Lines",
    toggle.control=TRUE
)
line.width.scale <- ol_scale_lwd_discrete(
    lwd.vector=c(
        vertical = 2,
        horizontal = 5
    ),
    name="Direction",
    display=TRUE
)
line.map.miami <- miami.gray.basemap +
    line.layer +
    line.width.scale
## Not run:
# Output to file and view
ol_map2HTML(
  line.map.miami,
  'miami_lines.html',
  map.heading="Miami Lines"
)
browseURL("miami_lines.html")

## End(Not run)
```

---

ol_scale_size_continuous

Point Size Scale (Continuous)

Description

Specify a continuous size mapping for a point layer.

Usage

```
ol_scale_size_continuous(low.val, high.val, low.size = 0.33,
  high.size = 0.75, name = NULL, na.size.val = 0.33,
  draw.fill = NULL, legend.breaks = NULL, display = FALSE)
```

Arguments

`low.val`	numeric the minimum variable value to be mapped to the smallest size.
`high.val`	numeric the maximum variable value to be mapped to the largest size.
`low.size`	numeric smallest size scalar.
`high.size`	numeric largest size scalar.
`name`	character the scale name.
`na.size.val`	numeric the size scalar assigned to non-numeric or NA values.
`draw.fill`	character fill color for drawing points in the legend only.
`legend.breaks`	numeric ordered vector of variable values to display in the legend.
`display`	logical indicating whether to draw the scale for output in the HTML. If `TRUE`, a bitmap will be created and sourced in the HTML in an `img` tag.

Details

This method maps OpenLayers point feature sizes to continuous variable values. This scale can be added to an Ol.Map S3 object only if the Ol.Map object has a layer with a "size" mapping to a numeric variable. If no such layer exists, attempts to add this type of scale will result in a warning. Attempt to apply this scale to a non-numeric variable will throw an error.

Size inputs to this method are applied as scalars to the icon widths. A size value of 1 translates to an icon width of 40 pixels for "pin" markers, or 20 pixels for "dot" markers.

Value

list of class `Scale.Size.Continuous`.

See Also

`ol_map`, `ol_geom_point`,

Examples

```
point.matrix <- matrix(
    c(
        -80.885+runif(10),
        25.223+runif(10)
    ),
    ncol=2
)
point.df <- data.frame(
    pt.type=sample(c("A","B"),10,replace=TRUE),
    pt.numeric=runif(10)*10
)
miami.points.map <- ol_map(
    center=c(-80.385790,25.782618),
    zoom=10
) +
    streetmap()+
    ol_geom_point(
        point.matrix,
        df=point.df,
        mapping=ol_aes(
            size=pt.numeric # continuous mapping
        ),
        name="Point",
        marker="pin",
        fill='green',
        toggle.control=TRUE,
        tooltip=sprintf("%1.2f",point.df$pt.numeric)
) +
    ol_scale_size_continuous(
    high.val=10,
    low.val=0,
    high.size=0.66,
    low.size=0.33,
    draw.fill='green',
    legend.breaks=c(0,3.33,6.67,10),
    display=TRUE
)
```

```
## Not run:
# Oave to file and open on browser
ol_map2HTML(
  miami.points.map,
  'pointsizes.html'
)
browseURL('pointsizes.html')

## End(Not run)
```

---

ol_scale_size_discrete

Point Size Scale (Discrete)

Description

Specify a discrete size mapping for a point layer.

Usage

```
ol_scale_size_discrete(size.vector = NULL, name = NULL,
  na.size.val = 0.33, draw.fill = NULL, legend.breaks = NULL,
  display = FALSE)
```

Arguments

`size.vector`	numeric named vector of the form `c(value=width)`. If `NULL`, a default size mapping is assigned.
`name`	character scale name.
`na.size.val`	numeric the size scalar assigned to unrecognized or NA values.
`draw.fill`	character fill color for drawing points in the legend only.
`legend.breaks`	numeric ordered vector of variable values to display in the legend.
`display`	logical indicating whether to draw the scale for output in the HTML. If `TRUE`, a bitmap will be created and sourced in the HTML in one or more `img` tags.

Details

This method maps OpenLayers point feature sizes to a discrete set of variable values. This scale can be added to an Ol.Map S3 object only if the Ol.Map object has a layer with a "size" mapping to a numeric, character, or factor variable. If no such layer exists, attempts to add this type of scale will result in a warning.

Size inputs to this method are applied as scalars to the icon widths. A size value of 1 translates to an icon width of 40 pixels for "pin" markers, or 20 pixels for "dot" markers.

Value

list of class `Scale.Size.Discrete`.

See Also

`ol_map`, `ol_geom_point`,

Examples

```
point.matrix <- matrix(
    c(
        -80.885+runif(10),
        25.223+runif(10)
    ),
    ncol=2
)
point.df <- data.frame(
    pt.type=sample(c("A","B"),10,replace=TRUE),
    pt.numeric=runif(10)*10
)
miami.points.map <- ol_map(
    center=c(-80.385790,25.782618),
    zoom=10
) +
    streetmap()+
    ol_geom_point(
        point.matrix,
        df=point.df,
        mapping=ol_aes(
            size=pt.type # continuous mapping
        ),
        name="Point",
        marker="pin",
        fill='green',
        toggle.control=TRUE,
        tooltip=point.df$pt.type
) +
    ol_scale_size_discrete(
    c(A=1,B=0.5),
    name="Point Type",
    draw.fill='black',
    display=TRUE
)
## Not run:
# Oave to file and open on browser
ol_map2HTML(
  miami.points.map,
  'pointsizes.html'
)
browseURL('pointsizes.html')

## End(Not run)
```

ROpenLayers *ROpenLayers: A pacakge for Geo-Visualization*

Description

ROpenLayers leverages the power of OpenLayers javascript libraries and web-based map servers to enable informative visualization.

What this package does

The purpose of this package is to make it easy for a user to visualize geo-spatial data and analyses using the open source OpenLayers javascript library and online map servers. The process for creating a visualization imitates the process of creating a plot in R package `ggplot2`.

1. First, an OpenLayers Map object is created with a call to the `ol_map` method.
2. Next, layers and scales are created and added. Layers can reference map servers to provide underlying base maps or vector features (polygons, lines, or points) created or imported in R. These capabilities are described in the following sections.
3. Finally, the updated map object is exported to HTML/javascript for viewing in a browser, hosting on a server, or embedding into another application or format. Export methods are `ol_map2HTML` and `ol_map2Strings`.

OpenLayers

OpenLayers is an open source javascript library that makes it easy to put a dynamic map on any web page. It is licensed under the 2-clause BSD license. This license will appear commented within OpenLayers CSS code in the HTML exports created by this package. However, this package does not contain any of the OpenLayers javascript source code; rather, it exports HTML code that source these libraries when loaded. Therefore, these products will not render without network access to the OpenLayers javascript library. By default, the products exported by this package source OpenLayers 3.16.0, but the user has the option to set the source URL (see `ol_map`).

ArcGIS Servers

The `user_arcgis_basemap` method allows the user to manually specify any available ArcGIS map server. This package also provides access to US National Geospatial-Intelligence Agency servers hosted at NGA.smil.mil through the `nga_basemap` method. Note that these servers require authentication, which will be requested at the time of access (i.e., when the HTML page is opened in a browser).

Vector Layers

This package enables users to rapidly access and write OpenLayers vector layers in javascript. The following methods enable that functionality.

- `ol_geom_polygon`
- `ol_geom_line`
- `ol_geom_point`
- `ol_geom_icon`
- `ol_geom_circle`
- `ol_geom_heatmap`
- `ol_geom_text`

Geocode

New in Version 0.0.9-99: geocoding addresses. See `geocode` for details.

Examples

```
data(quakes)
center <- c(mean(quakes$long),mean(quakes$lat))
quakes$long[which(quakes$long>180)]<-quakes$long[which(quakes$long>180)]-360
tooltips <- paste("Depth",quakes$depth,sep=": ")
mymap <- ol_map(
    zoom = 5,
    center = center
)
basemap.layer <- lightgray()
point.layer <- ol_geom_point(
    quakes[,c("long","lat")],
    mapping = ol_aes(fill=mag),
    df = quakes,
    name = "Earthquake Points",
    toggle.control=TRUE,
    tooltip = tooltips
)
heatmap.layer <- ol_geom_heatmap(
    quakes[,c("long","lat")],
    name = "Earthquake Heatmap",
    toggle.control=TRUE,
    weight.values = quakes$mag,
    opacity = 0.25
)
mymap <- mymap +
    basemap.layer +
    point.layer +
    ol_scale_fill_continuous(name="Magnitude",display=TRUE) +
    heatmap.layer
## Not run:
# Save to file and open in browser
ol_map2HTML(
  mymap,
  "Quakes.html",
  map.heading = "Earthquake Data Visualization"
)
browseURL("Quakes.html")

## End(Not run)
```

---

user_arcgis_basemap *User ArcGIS Basemap Layer*

Description

Create a basemap layer linking to an User-supplied ArcGIS mapserver.

Usage

```
user_arcgis_basemap(url, name = "", attributions = "",
  toggle.control = FALSE)
```

Arguments

url	character url string where the map server is located. Typically these urls end with "/MapServer".
name	character layer name.
attributions	character HTML. This HTML will render as attributional text at the bottom-right corner of the map. At a minimum, this text should include the copyright text provided on the map server.
toggle.control	logical. If TRUE, a checkbox will appear on the map allowing the viewer to toggle its visibility in the browser.

Details

Creates and returns an OpenLayers ArcGIS Tile layer that sources a map server at a user-supplied URL.

Value

A Layer.ArcGIS S3 object.

See Also

ol_map, +.Ol.Map, nga_basemap,

Examples

```
## NOTE: Example for public URL accessible on NIPR.
server.url <- "http://server.arcgisonline.com/arcgis/rest/services/NatGeo_World_Map/MapServer"
mymap <- ol_map()
attrib <- paste(
    "Content may not reflect National Geographic's current map policy.",
    "Sources: National Geographic, Esri, Garmin, HERE,",
    "UNEP-WCMC, USGS, NASA, ESA, METI, NRCAN, GEBCO, NOAA, increment P Corp",
    sep=" " # long attribution!
)
base.layer <- user_arcgis_basemap(
    server.url,
    attributions = attrib,
    toggle.control=TRUE
)
mymap <- mymap + base.layer
## Not run:
ol_map2HTML(
  mymap,
  "SanDiego_NatGeo.html",
  map.note = sprintf(
    "I found this at <a href='%s'>arcgisonline.com</a>",
    server.url
  )
)
browseURL("SanDiego_NatGeo.html")

## End(Not run)
```

Index