Package 'echarts'

July 17, 2017

Title EChart htmlwidget

Version 0.0.1

Description htmlwidget for ECharts 2.

Depends R (>= 3.3.2)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports htmlwidgets,
magrittr,
dplyr,
methods

RoxygenNote 6.0.1

URL http://john-coene.com/htmlwidgets/echarts/

BugReports https://github.com/JohnCoene/echarts/issues

R topics documented:

`eanimation` *Add animations*

Description

Add animations

Usage

```
eanimation(p, animation = TRUE, addDataAnimation = TRUE,
  animationThreshold = 2000, animationDuration = 2000,
  animationDurationUpdate = 500, animationEasing = "ExponentialOut", ...)
```

Arguments

p	an echart objects.
animation	whether to show the initial animation.
addDataAnimation	specifies whether the dynamic data interface animation will be enabled.
animationThreshold	threshold of animated elements.
animationDuration	duration of animation, in ms.
animationDurationUpdate	duration of the update animation, in ms.
animationEasing	easing effect, see details for valid values.
...	any other options.

Details

- linear
- QuadraticIn
- QuadraticOut
- QuadraticInOut
- CubicIn
- CubicOut
- CubicInOut
- QuarticIn
- QuarticOut
- QuarticInOut
- SinusoidalIn
- SinusoidalOut
- SinusoidalInOut
- ExponentialIn
- ExponentialOut
- ExponentialInOut
- CircularIn
- CircularOut
- CircularInOut

- ElasticIn
- ElasticOut
- ElasticInOut
- BackIn
- BackOut
- BackInOut
- BounceIn
- BounceOut
- BounceInOut

Examples

```
mtcars %>%
  echart(mpg) %>%
  ebar(qsec) %>%
  eanimation(animationEasing = "BounceIn")

mtcars %>%
  echart(mpg) %>%
  escatter(qsec, drat, symbolSize = 20) %>%
  eanimation(animationEasing = "CubicInOut")
```

---

earea *Add area*

Description

Add area serie.

Usage

```
earea(p, serie, name = NULL, stack = NULL, smooth = TRUE, ...)

earea_(p, serie, name = NULL, stack = NULL, smooth = TRUE, ...)
```

Arguments

p	an echart object.
serie	value column name to plot.
name	of serie.
stack	name of the stack.
smooth	whether to smooth line.
...	any other argument to pass to the serie. i.e.: same parameters as eline or eline_

Examples

```
df <- data.frame(x = LETTERS[1:10], y = runif(10, 30, 70), z = runif(10, 10, 50))

df %>%
  echart_("x") %>%
  earea_("y", smooth = FALSE, symbol = "emptyRectangle", symbolSize = 5)

df %>%
  echart(x) %>%
  earea(y, stack = "grp") %>%
  earea(z, stack = "grp") %>%
  etheme("roma")

df <- data.frame(x = 1:10, y = runif(10, 30, 70), z = runif(10, 10, 50))

df %>%
  echart(x) %>%
  earea(z, stack = "grp") %>%
  earea(y)
```

---

ebar *Add bars*

Description

Add bar serie.

Usage

```
ebar(p, serie, name = NULL, stack = NULL, clickable = TRUE,
  xAxisIndex = 0, yAxisIndex = 0, barGap = "100%",
  barCategoryGap = "20%", legendHoverLink = TRUE, z = 2, zlevel = 0,
  tooltip, ...)

ebar_(p, serie, name = NULL, stack = NULL, clickable = TRUE,
  xAxisIndex = 0, yAxisIndex = 0, barGap = "30%",
  barCategoryGap = "20%", legendHoverLink = TRUE, z = 2, zlevel = 0,
  tooltip, ...)
```

Arguments

p	an echart object.
serie	value column name to plot.
name	of serie.
stack	name of the stack.
clickable	whether plot is clickable.

`xAxisIndex, yAxisIndex`
: axis indexes.

`barGap, barCategoryGap`
: distance between each bar.

`legendHoverLink`
: enables legend hover links.

`z, zlevel`
: first and second grade cascading control, the higher z the closer to the top.

`tooltip`
: style of tooltip.

`...`
: any other argument to pass to the serie.

See Also

official bar options docs

Examples

```
mtcars %>%
  echart_("mpg") %>%
  ebar_("qsec")

mtcars %>%
  echart_("disp") %>%
  ebar_("mpg", stack = "grp") %>% # stack
  ebar_("qsec", stack = "grp") %>% # stack
  ebar_("wt", stack = "grp2") %>% # not stacked
  etooltip(trigger = "item") %>%
  elegend() %>%
  etoolbox_magic(type = list("stack", "tiled")) %>%
  etoolbox_restore()

df <- data.frame(x = LETTERS[1:4], y = runif(4, 0, 20), z = runif(4, 10, 15), w = runif(4, 15, 30))

df %>%
  echart(x) %>%
  ebar(y, stack = "grp") %>%
  ebar(z, stack = "grp") %>%
  ebar(w, "grp2") %>%
  etheme("macarons") %>%
  etooltip(trigger = "axis")
```

---

ecandle *Add candlestick*

Description

Add candlestick bars.

Usage

```
ecandle(p, opening, closing, low, high, name = NULL, clickable = TRUE,
  z = 2, zlevel = 0, ...)

ecandle_(p, opening, closing, low, high, name = NULL, clickable = TRUE,
  z = 2, zlevel = 0, ...)
```

Arguments

p	an echart object.
opening, closing, low, high	stock prices.
name	name of serie.
clickable	whether serie is clickable.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
...	any other options to pass to candlessticks.

See Also

candlestick official docs

Examples

```
# generate data
date <- c("2017-01-01", "2017-01-02", "2017-01-03", "2017-01-04", "2017-03-05",
          "2017-01-06", "2017-01-07")
stock <- data.frame(date = date,
                    opening = c(200.60, 200.22, 198.43, 199.05, 203.54, 203.40, 208.34),
                    closing = c(200.72, 198.85, 199.05, 203.73, 204.08, 208.11, 211.88),
                    low = c(197.82, 198.07, 197.90, 198.10, 202.00, 201.50, 207.60),
                    high = c(203.32, 200.67, 200.00, 203.95, 204.90, 208.44, 213.17))

stock %>%
  echart_("date") %>%
  ecandle_("opening", "closing", "low", "high")


js <- htmlwidgets::JS("function(params){
  var res = 'opening: ' + params.value[0] + '<br>' + 'closing: ' + params.value[3];
  return res}")

stock %>%
  echart(date) %>%
  ecandle(opening, closing, low, high, barMaxWidth = 20) %>%
  etooltip(trigger = "item", formatter = js) %>%
  etheme("macarons")
```

echart *Initiate an echart*

Description

Initiate an echart graph.

Usage

```
echart(data, x, width = NULL, height = NULL, elementId = NULL)

echart_(data, x, width = NULL, height = NULL, elementId = NULL)
```

Arguments

data	data.frame containing data to plot.
x	variable column.
width, height	dimensions of chart.
elementId	id of div containing chart.

echarts-shiny *Shiny bindings for echarts*

Description

Output and render functions for using echarts within Shiny applications and interactive Rmd documents.

Usage

```
echartsOutput(outputId, width = "100%", height = "400px")

renderEcharts(expr, env = parent.frame(), quoted = FALSE)
```

Arguments

outputId	output variable to read from
width, height	Must be a valid CSS unit (like '100%', '400px', 'auto') or a number, which will be coerced to a string and have 'px' appended.
expr	An expression that generates a echarts
env	The environment in which to evaluate expr.
quoted	Is expr a quoted expression (with quote())? This is useful if you want to save an expression in a variable.

echord *Add chord*

Description

Add chord chart.

Usage

```
echord(p, name = NULL, sort = "none", sortSub = "none",
  clickable = TRUE, z = 2, zlevel = 0, symbol = NULL,
  symbolSize = NULL, clockWise = FALSE, minRadius = 10, maxRadius = 20,
  ribbonType = TRUE, showScale = FALSE, showScaleText = FALSE,
  padding = 2, ...)

echord_(p, name = NULL, sort = "none", sortSub = "none",
  clickable = TRUE, z = 2, zlevel = 0, symbol = NULL,
  symbolSize = NULL, clockWise = FALSE, minRadius = 10, maxRadius = 20,
  ribbonType = TRUE, showScale = FALSE, showScaleText = FALSE,
  padding = 2, ...)
```

Arguments

p	an echart object.
name	name of serie.
sort, sortSub	data sorting, none, ascending or descending.
clickable	whether plot is clickable.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
symbol	marker, see details for valid values.
symbolSize	of symbol.
clockWise	whether links are displayed in clockwise direction.
minRadius, maxRadius	minimum and maximum radius after mapping to symbol size.
ribbonType	set to TRUE to use ribbons.
showScale	whether the scale will be showed. Only available if ribbonType is true.
showScaleText	whether to show scale text.
padding	distance between each sector.
...	any other options to pass to serie.

Details

Valid values for `symbol`:

- `circle`
- `rectangle`
- `triangle`
- `diamond`
- `emptyCircle`
- `emptyRectangle`
- `emptyTriangle`
- `emptyDiamond`
- `heart`
- `droplet`
- `pin`
- `arrow`
- `star`

See Also

official scatter options docs

Examples

```
set.seed(19880525)
matrix <- matrix(sample(0:1, 100, replace = TRUE, prob = c(0.9,0.6)), nc = 10)

matrix %>%
  echart_(LETTERS[1:10]) %>%
  echord_()

matrix %>%
  echart(LETTERS[1:10]) %>%
  echord(ribbonType = FALSE)
```

`ecloud` *Add wordcloud*

Description

Add wordcloud serie.

Usage

```
ecloud(p, freq, color, name = NULL, clickable = TRUE,
  center = list("50%", "50%"), size = list("40%", "40%"),
  textRotation = list(0, 90), autoSize = list(enable = TRUE, minSize = 12),
  z = 2, zlevel = 0, tooltip, ...)

ecloud_(p, freq, color = NULL, name = NULL, clickable = TRUE,
  center = list("50%", "50%"), size = list("40%", "40%"),
  textRotation = list(0, 90), autoSize = list(enable = TRUE, minSize = 12),
  z = 2, zlevel = 0, tooltip, ...)
```

Arguments

p	an echart object.
freq	frequencies.
color	color of terms.
name	name of wordcloud.
clickable	whether terms are clickable.
center	center of cloud.
size	size of cloud.
textRotation	horizontal and vertical text rotation.
autoSize	automatic text size computation.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
tooltip	cutomise tooltip.
...	any other argument to pass to funnel.

See Also

official wordcloud docs

Examples

```
tf <- data.frame(terms = c("ECharts", "htmlwidgets", "rstats", "htmltools"),
  freq = c(20, 17, 15, 7), color = c("red", "orange", "yellow", "grey"))

tf %>%
  echart_("terms") %>%
  ecloud_("freq", "color") %>%
  etooltip()
```

ecolorbar *Customise colorbar*

Description

Customise the colorbar of your chart.

Usage

```
ecolorbar(p, min = NULL, max = NULL, which = "previous", show = TRUE,
  color = NULL, zlevel = 4, z = 0, orient = "vertical", x = "left",
  y = "bottom", backgroundColor = "rgba(0,0,0,0)", borderColor = "#ccc",
  borderWidth = 0, padding = 5, itemGap = 10, itemWidth = 20,
  itemHeight = 14, precision = 0, splitNumber = 5, splitList = NULL,
  range = NULL, selectedMode = TRUE, calculable = FALSE,
  hoverLink = TRUE, realtime = FALSE, ...)
```

Arguments

p	an echart object.
min, max	minimum and maximum.
which	series to serie is to be affected, takes the name of a serie, previous or all.
show	whether to show the color bar.
color	colors as list from high to low. i.e.: list("red", "blue").
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
orient	orientation of bar, vertical or horizontal.
x	x position; left or right.
y	y posotion; top or bottom.
backgroundColor	background color.
borderColor	border color.
borderWidth	width of border.
padding	padding.
itemGap	gap between items on bar.
itemWidth	width of the bar.
itemHeight	height of the bar.
precision	decimal precision.
splitNumber	number of segments.
splitList	see official docs for details.
range	used to set initial range i.e.: list(start = 10, end = 50).
selectedMode	selection mode.

calculable	whether values are calculable.
hoverLink	hoverlink with map.
realtime	set to TRUE if using real time stream.
...	any other argument to pass to color bar.

Details

ecolorbar refers to datarange in docs.

See Also

official dataRange docs

Examples

```
df <- data.frame(x = 1:20,
                 y = runif(20, 5, 10),
                 size = runif(20, 5, 15))

df %>%
  echart(x) %>%
  escatter(y, size, symbolSize = 10, legendHoverLink = TRUE) %>%
  ecolorbar(color = list("red", "blue"), min = 5, max = 15, calculable = TRUE)
```

---

edata *Add data*

Description

Add a dataset.

Usage

```
edata(p, data, x)

edata_(p, data, x)
```

Arguments

p	an echart object.
data	data.frame.
x	x variable.

See Also

emap

`eforce` *Build force network*

Description

Build force network

Plot force directed graph.

Usage

```
eforce(p, name = NULL, large = FALSE, center = list("50%", "50%"),
  roam = FALSE, size = "100%", ribbonType = FALSE, minRadius = 10,
  maxRadius = 20, linkSymbol = "none", linkSymbolSize = list(10, 15),
  scaling = 1, gravity = 1, draggable = TRUE, useWorker = TRUE,
  steps = 1, z = 2, zlevel = 0, ...)

eforce_(p, name = NULL, large = FALSE, center = list("50%", "50%"),
  roam = FALSE, size = "100%", ribbonType = FALSE, minRadius = 10,
  maxRadius = 20, linkSymbol = "none", linkSymbolSize = list(10, 15),
  scaling = 1, gravity = 1, draggable = TRUE, useWorker = TRUE,
  steps = 1, z = 2, zlevel = 0, ...)
```

Arguments

`p`	an echart objects.
`name`	name of network.
`large`	set to `TRUE` to optimise for large graphs.
`center`	center of network.
`roam`	set to `TRUE` to enable zoom and drag.
`size`	size of layout.
`ribbonType`	whether to use ribbons.
`minRadius, maxRadius`	minimum and maximum radius of nodes.
`linkSymbol`	can be set to `arrow`.
`linkSymbolSize`	size of `symbol`.
`scaling`	scaling factor.
`gravity`	centripetal force coefficient.
`draggable`	set to `TRUE` to allow dragging nodes.
`useWorker`	specifies whether to put layout calculation into web worker when the browser supports web worker.
`steps`	the number of iterations of each frame layout calculation.
`z, zlevel`	first and second grade cascading control, the higher z the closer to the top.
`...`	any other options to pass to serie.

See Also

enodes eforce

Examples

```
let <- LETTERS[1:20]

edges <- data.frame(source = sample(let, 20), target = sample(let, 20),
  weight = runif(20, 5, 20))

nodes <- data.frame(name = let, value = runif(20, 5, 25), group = rep(LETTERS[1:4], 5))

echart() %>%
  eforce(itemStyle = list(normal = list(label = list(show = TRUE)))) %>% # show labels
  enodes(nodes, name, value = value, category = group) %>%
  elinks(edges, source, target)
```

---

efunnel *Add funnel*

Description

Add funnel

Usage

```
efunnel(p, serie, name = NULL, clickable = TRUE, legendHoverLink = TRUE,
  sort = "descending", min = NULL, max = NULL, x = 80, y = 60,
  x2 = 80, y2 = 60, width = NULL, height = NULL,
  funnelAlign = "center", minSize = "0%", maxSize = "100%", gap = 0,
  tooltip, ...)

efunnel_(p, serie, name = NULL, clickable = TRUE, legendHoverLink = TRUE,
  sort = "descending", min = 0, max = NULL, x = 80, y = 60, x2 = 80,
  y2 = 60, width = NULL, height = NULL, funnelAlign = "center",
  minSize = "0%", maxSize = "100%", gap = 0, tooltip, ...)
```

Arguments

p	an echart object.
serie	values to plot.
name	name of serie.
clickable	whether segments are clickable.
legendHoverLink	enables legend hover link.

sort	data sorting, takes descending or ascending.
min, max	minimum and maximum values of funnel.
x, y, x2, y2	coordinates of funnel.
width, height	width and height of funnel.
funnelAlign	alignment of funnel takes left, right and center.
minSize, maxSize	minimum and maximum size of funnel.
gap	gap between segments.
tooltip	cutomise tooltip.
...	any other argument to pass to funnel.

See Also

official funnel docs

Examples

```
funnel <- data.frame(stage = c("View", "Click", "Purchase"), value = c(80, 30, 20))

funnel %>%
  echart_("stage") %>%
  efunnel_("value")
```

---

egauge *Add gauge*

Description

Add gauge.

Usage

```
egauge(p, value, indicator = "", name = NULL, clickable = FALSE,
  legendHoverLink = TRUE, center = list("50%", "50%"),
  radius = list("0%", "75%"), startAngle = 225, endAngle = -45,
  min = 0, max = 100, splitNumber = 10, z = 2, zlevel = 0, tooltip,
  ...)

egauge_(p, value, indicator = "", name = NULL, clickable = FALSE,
  legendHoverLink = TRUE, center = list("50%", "50%"),
  radius = list("0%", "75%"), startAngle = 225, endAngle = -45,
  min = 0, max = 100, splitNumber = 10, z = 2, zlevel = 0, tooltip,
  ...)
```

Arguments

p	an echart object.
value	value to plot.
indicator	indicator appearing in center of gauge.
name	name of serie.
clickable	whether the item is clickable.
legendHoverLink	enables legend hover link.
center	center of gauge in pixels of percent.
radius	radius of gauge in pixels of percent.
startAngle, endAngle	start and end angles of gauge.
min, max	minimum and maximum of gauge.
splitNumber	number of segments.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
tooltip	customise tooltip.
...	any other arguments.

See Also

official gauge docs

Examples

```
echart() %>%
  egauge(85, "SPEED")

echart() %>%
  egauge(25, "SPEED") %>%
  etheme("helianthus")

echart() %>%
  egauge(63, "PERCENT") %>%
  etheme("dark")
```

---

egrid *Customise grid*

Description

Customise grid

Usage

```
egrid(p, backgroundColor = NULL, borderWidth = 1, borderColor = NULL,
  width = NULL, height = NULL, z = 0, zlevel = 0, x = 80, y = 60,
  x2 = 80, y2 = 80)
```

Arguments

p	an echart object.
backgroundColor	background color.
borderWidth	border width.
borderColor	border color.
width, height	dimensions of grid.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
x, y	ordinate on upper left corner.
x2, y2	ordinate on upper right corner.

See Also

official grid docs

Examples

```
df <- data.frame(x = 1:20, y = runif(20, 5, 20))

df %>%
  echart(x) %>%
  eline(x) %>%
  egrid(borderWidth = 5, borderColor = "red", backgroundColor = "yellow")
```

---

eheatmap *Add heatmap*

Description

Add heatmap.

Usage

```
eheatmap(p, y, values, name = NULL, clickable = TRUE, blurSize = 30,
  minAlpha = 0.5, valueScale = 1, opacity = 1, z = 2, zlevel = 0,
  gradientColors, tooltip, ...)

eheatmap_(p, y, values, name = NULL, clickable = TRUE, blurSize = 30,
  minAlpha = 0.5, valueScale = 1, opacity = 1, z = 2, zlevel = 0,
  gradientColors, tooltip, ...)
```

Arguments

p	an echart object.
y	yaxis values.
values	heat.
name	name of serie.
clickable	whether chart is clickable.
blurSize	size of points blur.
minAlpha	minimum transparency.
valueScale	values multiplier.
opacity	opacity of heatmap.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
gradientColors	colors used for gradient as a list i.e.:list("red", "blue")
tooltip	cutomise tooltip.
...	any other options to pass to heatmap.

See Also

official heatmap docs

Examples

```
set.seed(19880525)
matrix <- data.frame(x = runif(150, 10, 500), y = runif(150, 10, 500), z = runif(150, 10 , 200))

matrix %>%
  echart_("x") %>%
  eheatmap_("y", "z")
```

---

elegend *Add legend*

Description

Add legend

Usage

```
elegend(p, legend, show = TRUE, zlevel = 0, z = 4,
  orient = "horizontal", x = "center", y = "top",
  backgroundColor = "rgba(0,0,0,0)", borderColor = "#ccc",
  borderWidth = 0, padding = 5, itemGap = 10, itemWidth = 20,
  itemHeight = 14, formatter = NULL, selectedMode = TRUE,
  selected = NULL, textStyle, ...)
```

Arguments

p	an echart object.
legend	legend.
show	wether to show legend.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
orient	orientation, vertical or horizontal.
x	x alignment, center, left or right.
y	y alignment, center, top or bottom.
backgroundColor	background color.
borderColor	border color.
borderWidth	border width.
padding	legend padding.
itemGap	gap between legend items.
itemWidth, itemHeight	width and height of items.
formatter	default formatter.
selectedMode	selection mode.
selected	default selected state.
textStyle	textStyle.
...	any other option to pass to legend.

Examples

```
df <- data.frame(x = LETTERS[1:10], y = runif(10, 0, 10), z = runif(10, 0, 10))

df %>%
  echart(x) %>%
  ebar(y, name = "y - serie") %>%
  ebar(z) %>%
  elegend()
```

---

eline *Add lines*

Description

Add line serie.

Usage

```
eline(p, serie, name = NULL, stack = NULL, clickable = TRUE,
  xAxisIndex = 0, yAxisIndex = 0, symbol = NULL, symbolSize = "2 | 4",
  symbolRotate = NULL, showAllSymbol = FALSE, smooth = TRUE,
  legendHoverLink = TRUE, dataFilter = "nearest", z = 2, zlevel = 0,
  tooltip, ...)

eline_(p, serie, name = NULL, stack = NULL, clickable = TRUE,
  xAxisIndex = 0, yAxisIndex = 0, symbol = NULL, symbolSize = "4",
  symbolRotate = NULL, showAllSymbol = FALSE, smooth = TRUE,
  legendHoverLink = TRUE, dataFilter = "nearest", z = 2, zlevel = 0,
  tooltip, ...)
```

Arguments

p	an echart object.
serie	value column name to plot.
name	of serie.
stack	name of the stack.
clickable	whether plot is clickable.
xAxisIndex, yAxisIndex	axis indexes.
symbol	symbol for point marker, see details for valid values.
symbolSize	of symbol.
symbolRotate	angle by which symbol is rotated, i.e.: 30.
showAllSymbol	By default, a symbol will show only when its corresponding axis label does.
smooth	whether to smooth line.
legendHoverLink	enables legend hover link to the chart.
dataFilter	ECharts data filtering strategy, see details.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
tooltip	style of tooltip.
...	any other argument to pass to the serie.

Details

Valid values for symbol:

- circle
- rectangle
- triangle
- diamond
- emptyCircle

- emptyRectangle
- emptyTriangle
- emptyDiamond
- heart
- droplet
- pin
- arrow
- star

dataFilter: ECharts will optimize for the situation when data number is much larger than viewport width. It will filter the data showed in one pixel width. And this option is for data filtering strategy.

Valid values for dataFilter are:

- nearest (default)
- min
- max
- average

See Also

official line options docs

Examples

```
df <- data.frame(x = 1:50, y = runif(50, 5, 10), z = runif(50, 7, 12), w = runif(50, 10, 13))

df %>%
  echart(x) %>%
  eline(y) %>%
  eline(z)

# JS sizing function
sizing <- htmlwidgets::JS("function(value){ return value[1]/1.5}")

df %>%
  echart_("x") %>%
  eline_("y", "w",
         symbolSize = sizing,
         showAllSymbol = TRUE,
         symbol = "emptyCircle") %>%
  etooltip() %>%
  etheme("helianthus")

df %>%
  echart_("x") %>%
  eline_("y", stack = "grp") %>%
 eline_("z", stack = "grp", symbol = "emptyDroplet", showAllSymbol = TRUE, symbolSize = 5) %>%
 eline_("w", showAllSymbol = TRUE, symbolSize = 4, symbol = "emptyHeart", stack = "grp2") %>%
```

```
  etooltip() %>%
  elegend() %>%
  etoolbox_magic(type = list("line", "bar"))
```

---

elinks *Add edges*

Description

Add edges for eforce.

Add edges for eforce.

Usage

```
elinks(p, links, source, target, weight = 1)

elinks_(p, links, source, target, weight = 1)
```

Arguments

p	an echart object.
links	edges data.frame.
source	source column.
target	target column.
weight	edge weight.

See Also

enodes eforce

Examples

```
let <- LETTERS[1:20]

edges <- data.frame(source = sample(let, 20), target = sample(let, 20),
  weight = runif(20, 5, 20))

nodes <- data.frame(name = let, value = runif(20, 5, 25), group = rep(LETTERS[1:4], 5))

echart() %>%
  eforce(itemStyle = list(normal = list(label = list(show = TRUE)))) %>% # show labels
  enodes(nodes, name, value = value, category = group) %>%
  elinks(edges, source, target)

echart() %>%
  eforce(itemStyle = list(normal = list(label = list(show = TRUE)))) %>% # show labels
  enodes(nodes, name, value = value, category = group) %>%
```

```
  elinks(edges, source, target, weight = 1)

let <- LETTERS[1:20]

edges <- data.frame(source = sample(let, 20), target = sample(let, 20),
  weight = runif(20, 5, 20))

nodes <- data.frame(name = let, value = runif(20, 5, 25), group = rep(LETTERS[1:4], 5))

echart() %>%
  eforce_(itemStyle = list(normal = list(label = list(show = TRUE)))) %>% # show labels
  enodes_(nodes, "name", value = "value", category = "group") %>%
  elinks_(edges, "source", "target")
```

---

emap *Add blank map*

Description

Setup map plot.

Usage

```
emap(p, name = NULL, mapType = "world", clickable = TRUE, z = 2,
  zlevel = 0, selectedMode = NULL, hoverable = FALSE,
  dataRangeHoverLink = TRUE, mapLocation = list(x = "center", y = "center"),
  mapValueCalculation = "sum", mapValuePrecision = 0,
  showLegendSymbol = TRUE, roam = FALSE, scaleLimit = NULL,
  nameMap = NULL, textFixed = NULL, ...)

emap_(p, name = NULL, mapType = "world", clickable = TRUE, z = 2,
  zlevel = 0, selectedMode = NULL, hoverable = FALSE,
  dataRangeHoverLink = TRUE, mapLocation = list(x = "center", y = "center"),
  mapValueCalculation = "sum", mapValuePrecision = 0,
  showLegendSymbol = TRUE, roam = FALSE, scaleLimit = NULL,
  nameMap = NULL, textFixed = NULL, ...)
```

Arguments

p	an echart object.
name	name of serie.
mapType	type of map, see examples.
clickable	whether elements are clickable.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
selectedMode	whether items can be selected.

hoverable	whether elements are hoverable.
dataRangeHoverLink	enables dataRange hover link to the chart.
mapLocation	x and y location of map on canvas, takes top, bottom, left, right, center.
mapValueCalculation	takes sum or average.
mapValuePrecision	decimal precision.
showLegendSymbol	whether to show symbol on legend.
roam	enables zoom and drag.
scaleLimit	controls drag and zoom limits.
nameMap	custom name mapping.
textFixed	fixed text location for a region.
...	any other options to pass to map serie.

See Also

emap_coords, emap_heat, emap_lines, emap_choropleth, emap_points, official map docs

Examples

```
coords <- data.frame(city = c("London", "New York", "Beijing", "Sydney"),
  lon = c(-0.1167218, -73.98002, 116.3883, 151.18518),
  lat = c(51.49999, 40.74998, 39.92889, -33.92001),
  values = runif(4, 10, 20))

coords %>%
  echart_("city") %>% # initialise chart
  emap_() %>% # setup default map
  emap_coords_("lon", "lat") %>% # add coordinates
  emap_points_("values") # plot values on coordinates

edges <- data.frame(source = c("Beijing", "Beijing", "New York"),
  target = c("Sydney", "London", "London"))

coords %>%
  echart_("city") %>%
  emap_() %>%
  emap_coords_("lon", "lat") %>%
  emap_lines_(edges, "source", "target")

data <- data.frame(lon = runif(200, 90, 120),
  lat = runif(200, 30, 39),
  z = runif(200, 50, 75))

data %>%
  echart_() %>%
  emap_(mapType = "china") %>%
```

```
  emap_heat_("lon", "lat", "z")

us_data <- data.frame(state = c("New York", "Los Angeles", "Dallas"),
                      lat = c(40.730610, 34.052235, 33.940369),
                      lon = c(-73.935242, -118.243683, -84.692894),
                      values = round(runif(3, 1, 2)))

us_data %>%
  echart_("state") %>%
  emap(mapType = "world|United States of America") %>%
  emap_coords_("lon", "lat") %>%
  emap_points_("values")
```

`emap_choropleth` *Add choropleth*

Description

Add choropleth

Usage

```
emap_choropleth(p, serie)

emap_choropleth_(p, serie)
```

Arguments

p	an echart object.
serie	values to plot.

See Also

ecolorbar

Examples

```
choropleth <- data.frame(countries = c("France", "Brazil", "China", "Russia", "Canada", "India"),
  values = round(runif(6, 10, 25)))

choropleth %>%
  echart_("countries") %>%
  emap_() %>%
  emap_choropleth_("values")

choropleth %>%
  echart_("countries") %>%
  emap() %>%
```

```
  emap_choropleth(values) %>%
  ecolorbar(color = list("red", "yellow"), calculable = TRUE)
```

---

emap_coords *Add map coordinates*

Description

Add coordinates to map.

Usage

```
emap_coords(p, lon, lat)

emap_coords_(p, lon, lat)
```

Arguments

p	an echart object
lon, lat	coordinates to plot.

Examples

```
coords <- data.frame(city = c("London", "New York", "Beijing", "Sydney"),
  lon = c(-0.1167218, -73.98002, 116.3883, 151.18518),
  lat = c(51.49999, 40.74998, 39.92889, -33.92001))

edges <- data.frame(source = c("Beijing", "Beijing", "New York"),
  target = c("Sydney", "London", "London"))

coords %>%
  echart_("city") %>%
  emap() %>%
  emap_coords_("lon", "lat") %>%
  emap_lines_(edges, "source", "target")
```

emap_heat *Add heat on map*

Description

Add heat on map

Usage

```
emap_heat(p, lon, lat, z, blurSize = 30, minAlpha = 0.05, valueScale = 1,
  opacity = 1, gradientColors = NULL, ...)

emap_heat_(p, lon, lat, z, blurSize = 30, minAlpha = 0.05, valueScale = 1,
  opacity = 1, gradientColors = NULL, ...)
```

Arguments

p	an echart object.
lon, lat	coordinates.
z	values, heat.
blurSize	blur of points.
minAlpha	minimum transparency.
valueScale	z multiplier.
opacity	opacity of heatmap.
gradientColors	colors.
...	any other parameter to pass to heatmap.

Examples

```
data <- data.frame(lon = runif(300, 90, 120),
  lat = runif(300, 30, 39),
  z = runif(300, 75, 100))

data %>%
  echart_() %>%
  emap(mapType = "china") %>%
  emap_heat_("lon", "lat", "z")

data %>%
  echart() %>%
  emap(mapType = "china") %>%
  emap_heat_("lon", "lat", "z", blurSize = 50, minAlpha = 0.3, opacity = 0.8)
```

emap_lines *Add map lines*

Description

Add lines on map.

Usage

```
emap_lines(p, edges, source, target, name = NULL, clickable = TRUE,
  symbol = "arrow", symbolSize = 2, symbolRotate = NULL, large = FALSE,
  smooth = TRUE, z = 2, zlevel = 0, smoothness = 0.2, precision = 2,
  bundling = list(enable = FALSE, maxTurningAngle = 45), ...)

emap_lines_(p, edges, source, target, name = NULL, clickable = TRUE,
  symbol = "arrow", symbolSize = 2, symbolRotate = NULL, large = FALSE,
  smooth = TRUE, z = 2, zlevel = 0, smoothness = 0.2, precision = 2,
  bundling = list(enable = FALSE, maxTurningAngle = 45), ...)
```

Arguments

p	an echart object.
edges	edges data.frame.
source, target	source and target columns in edges data.frame.
name	name of serie.
clickable	whether lines are clikable.
symbol	symbol, see valid details for valid values.
symbolSize	of symbol.
symbolRotate	angle by which symbol is rotated, i.e.: 30.
large	optimises for 2'000 data points and over.
smooth	whether to smooth lines.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
smoothness	line smoothness
precision	for 'average'.
bundling	edge bundling settings, see usage.
...	any other options to pass to line.

Details

Valid values for symbol:

- circle
- rectangle

- triangle
- diamond
- emptyCircle
- emptyRectangle
- emptyTriangle
- emptyDiamond
- heart
- droplet
- pin
- arrow
- star

See Also

emap_coords official map line docs

Examples

```
coords <- data.frame(city = c("London", "New York", "Beijing", "Sydney"),
  lon = c(-0.1167218, -73.98002, 116.3883, 151.18518),
  lat = c(51.49999, 40.74998, 39.92889, -33.92001))

edges <- data.frame(source = c("Beijing", "Beijing", "New York"),
  target = c("Sydney", "London", "London"))

coords %>%
  echart_("city") %>%
  emap() %>%
  emap_coords_("lon", "lat") %>%
  emap_lines_(edges, "source", "target")

edges2 <- data.frame(source = "London", target = "Sydney")

coords %>%
  echart_("city") %>%
  emap() %>%
  emap_coords_("lon", "lat") %>%
  emap_lines_(edges, "source", "target") %>%
  emap() %>%
  emap_coords_("lon", "lat") %>%
  emap_lines_(edges2, "source", "target", effect = emap_line_effect()) %>%
  etheme("helianthus")

coords2 <- data.frame(city = "Sydney", lon = 151.18518, lat = -33.92001, value = 20)

coords %>%
  echart_("city") %>%
  emap() %>%
```

```
  emap_coords_("lon", "lat") %>%
  emap_lines_(edges, "source", "target") %>%
  edata_(coords2, "city") %>%
  emap() %>%
  emap_coords_("lon", "lat") %>%
  emap_lines_(edges2, "source", "target", effect = emap_line_effect(scaleSize = 2)) %>%
  emap_coords_("lon", "lat") %>%
 emap_points_("value", symbol = "emptyCircle", effect = list(show = TRUE, shadowBlur = 10)) %>%
  etheme("dark")
```

`emap_line_effect` *emap line effect*

Description

Effect for emap lines

Usage

```
emap_line_effect(show = TRUE, loop = TRUE, period = 30, scaleSize = 1,
  color = "#fff", shadowBlur = 10, shadowColor = NULL, ...)
```

Arguments

`show`	set to `TRUE` to show effect.
`loop`	set to `TRUE` to loop animation.
`period`	period loop.
`scaleSize`	scale.
`color`	color.
`shadowBlur`	blur.
`shadowColor`	color of shadow.
`...`	any other option to pass to effect.

Examples

```
coords <- data.frame(city = c("London", "New York", "Beijing", "Sydney"),
  lon = c(-0.1167218, -73.98002, 116.3883, 151.18518),
  lat = c(51.49999, 40.74998, 39.92889, -33.92001))

edges <- data.frame(source = c("Beijing", "Beijing", "New York"),
  target = c("Sydney", "London", "London"))

coords %>%
  echart(city) %>%
  emap() %>%
  emap_coords(lon, lat) %>%
  emap_lines(edges, source, target, effect = emap_line_effect())
```

`emap_points` *Add map points*

Description

Add map points

Usage

```
emap_points(p, serie, clickable = TRUE, symbol = "pin", symbolSize = 10,
  symbolRotate = NULL, large = FALSE, itemStyle = NULL, ...)

emap_points_(p, serie, clickable = TRUE, symbol = "pin", symbolSize = 10,
  symbolRotate = NULL, large = FALSE, itemStyle = NULL, ...)
```

Arguments

`p`	an echart objects.
`serie`	values to plot.
`clickable`	whether points are clickable.
`symbol`	point symbol, see details for valid values.
`symbolSize`	size of points.
`symbolRotate`	angle by which symbol is rotated, i.e.: `30`.
`large`	whether to optimise for large datasets: 2K points +.
`itemStyle`	cutomise points.
`...`	any other option to pass to points.

Details

Valid values for `symbol`:

- `circle`
- `rectangle`
- `triangle`
- `diamond`
- `emptyCircle`
- `emptyRectangle`
- `emptyTriangle`
- `emptyDiamond`
- `heart`
- `droplet`
- `pin`
- `arrow`
- `star`

See Also

office map points docs

Examples

```
coords <- data.frame(city = c("London", "New York", "Beijing", "Sydney"),
  lon = c(-0.1167218, -73.98002, 116.3883, 151.18518),
  lat = c(51.49999, 40.74998, 39.92889, -33.92001),
  values = runif(4, 10, 20))

coords %>%
  echart_("city") %>%
  emap() %>%
  emap_coords_("lon", "lat") %>%
  emap_points_("values")

coords2 <- data.frame(city = "Rio", lon = -43.172896, lat = -22.906847, value = 15)

coords %>%
  echart_("city") %>%
  emap() %>%
  emap_coords_("lon", "lat") %>%
  emap_points_("values", symbolSize = 5) %>%
  edata_(coords2, "city") %>%
  emap() %>%
  emap_coords_("lon", "lat") %>%
 emap_points_("value", symbol = "emptyCircle", effect = list(show = TRUE, shadowBlur = 10)) %>%
  etheme("helianthus")
```

---

emap_roam *Add Zoom and roam controller*

Description

Add zoom and roam controller to map.

Usage

```
emap_roam(p, show = TRUE, zlevel = 0, z = 4, x = "left", y = "top",
  width = 80, height = 120, backgroundColor = "rgba(0,0,0,0)",
  borderColor = "#ccc", borderWidth = 0, padding = 5,
  fillerColor = "#fff", handleColor = "#6495ed", step = 15,
  mapTypeControl = NULL, ...)
```

Arguments

p	an echart object.
show	set to TRUE to show the controller
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
x	x position; left or right.
y	y posotion; top or bottom.
width, height	dimensions of controller.
backgroundColor	background color.
borderColor	border color.
borderWidth	width of border.
padding	padding.
fillerColor	filler color.
handleColor	color of handle.
step	moving step of the 4 direction roam in px.
mapTypeControl	ou can specify every single mapType when multiple map in a chart at the same time, such as: list({ china = FALSE, world = TRUE}).
...	any other option to pass to controller.

See Also

official roam controller docs

Examples

```
coords <- data.frame(city = c("London", "New York", "Beijing", "Sydney"),
  lon = c(-0.1167218, -73.98002, 116.3883, 151.18518),
  lat = c(51.49999, 40.74998, 39.92889, -33.92001),
  values = runif(4, 10, 20))

coords %>%
  echart_("city") %>%
  emap() %>%
  emap_coords_("lon", "lat") %>%
  emap_points_("values") %>%
  emap_roam(mapTypeControl = list(world = TRUE))
```

emark_line *mark line*

Description

mark line

Usage

```
emark_line(p, which = "previous", data = list(), clickable = TRUE,
  symbol = list("circle", "arrow"), symbolSize = list(2, 4),
  symbolRotate = NULL, large = FALSE, smooth = FALSE, precision = 2,
  bundling = list(enable = FALSE, maxTurningAngle = 45), z = 2,
  zlevel = 0, ...)
```

Arguments

p	an echart object.
which	serie to mark lines of, takes previous, all or name of specific serie.
data	data of mark points, points to mark.
clickable	whether marked points are clickable.
symbol	symbol, see details for valid values.
symbolSize	size of symbol.
symbolRotate	symbol rotation angle, i.e.:30.
large	set to TRUE to optimise for large datasets.
smooth	whether to smooth line.
precision	decimal precision.
bundling	line bundling.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
...	any other options to pass to mark points.

Details

Valid values for symbol:

- circle
- rectangle
- triangle
- diamond
- emptyCircle
- emptyRectangle
- emptyTriangle

- emptyDiamond
- heart
- droplet
- pin
- arrow
- star

Examples

```
df <- data.frame(x = c("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"),
                 radio = c(320, 332, 301, 334, 390, 330, 320),
                 social = c(120, 132, 101, 134, 90, 230, 210))

df %>%
  echart(x) %>%
  ebar(radio, stack = "grp") %>%
  ebar(social, stack = "grp") %>%
  emark_line(data = list(list(type = "max")), clickable = FALSE) %>%
  etooltip(axisPointer = list(type = "shadow")) %>%
  etoolbox_full() %>%
  etoolbox_magic(type = list("tiled", "stack", "line", "bar"))
```

---

emark_point *mark points*

Description

mark points

Usage

```
emark_point(p, which = "previous", data = list(), clickable = TRUE,
  symbol = "pin", symbolSize = 10, symbolRotate = NULL, large = FALSE,
  ...)
```

Arguments

p	an echart object.
which	serie to mark lines of, takes previous, all or name of specific serie.
data	data of mark points, points to mark.
clickable	whether marked points are clickable.
symbol	symbol, see details for valid values.
symbolSize	size of symbol.
symbolRotate	symbol rotation angle, i.e.:30.
large	set to TRUE to optimise for large datasets.
...	any other options to pass to mark points.

Details

Valid values for `symbol`:

- `circle`
- `rectangle`
- `triangle`
- `diamond`
- `emptyCircle`
- `emptyRectangle`
- `emptyTriangle`
- `emptyDiamond`
- `heart`
- `droplet`
- `pin`
- `arrow`
- `star`

Examples

```
df <- data.frame(x = 1:150,
                 y = round(rnorm(150, mean = 5, sd = 1), 2),
                 z = round(rnorm(150, mean = 7, sd = 1), 2))

df %>%
  echart(x) %>%
  escatter(y) %>%
  escatter(z) %>%
  emark_point(which = "all", data = list(list(type = "min"), list(type = "max")))
```

`eoptions` *Add global options*

Description

Add global options.

Usage

```
eoptions(p, backgroundColor = NULL, renderAsImage = FALSE,
  calculable = FALSE, color = NULL, symbolList = NULL, ...)
```

Arguments

p	an echart object.
backgroundColor	background color.
renderAsImage	allows rendering as image.
calculable	specifies whether the drag-recalculate feature will be enabled.
color	colors to use in chart.
symbolList	list of default symbols.
...	any other options.

Examples

```
mtcars %>%
  echart(mpg) %>%
  eline(qsec) %>%
  eoptions(backgroundColor = "black")
```

---

epie *Add pie*

Description

Add pie chart

Usage

```
epie(p, serie, name = NULL, clickable = TRUE, legendHoverLink = TRUE,
  center = list("50%", "50%"), radius = list(0, "75%"),
  startAngle = 90, minAngle = 0, clockWise = TRUE, roseType = NULL,
  selectedOffset = 10, selectedMode = TRUE, z = 2, zlevel = 0, ...)

epie_(p, serie, name = NULL, clickable = TRUE, legendHoverLink = TRUE,
  center = list("50%", "50%"), radius = list(0, "75%"),
  startAngle = 90, minAngle = 0, clockWise = TRUE, roseType = NULL,
  selectedOffset = 10, selectedMode = TRUE, z = 2, zlevel = 0, ...)
```

Arguments

p	an echart object.
serie	value column name to plot.
name	of serie.
clickable	whether plot is clickable.
legendHoverLink	enables legend hover links.

center	coordinates of the center.
radius	radius in pixels or percent.
startAngle, minAngle	start and minimum angle.
clockWise	whether to display slices in clockwise direction
roseType	type of pie, takes NULL, area or radius, see examples.
selectedOffset	offset of selected slice.
selectedMode	whether slices are selectable.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
...	any other option to pass to serie.

See Also

official pie options docs

Examples

```
pie <- data.frame(name = c("banana", "apple", "pineapple", "onion"),
  value = c(26, 15, 12, 9))

pie %>%
  echart_("name") %>%
  epie(value)

pie %>%
  echart(name) %>%
  epie(value, roseType = "area") %>%
  etheme("helianthus")

pie %>%
  echart_("name") %>%
  epie_("value", roseType = "radius") %>%
  etheme("blue")
```

---

eradar *Add radar*

Description

Add radar chart.

Usage

```
eradar(p, serie, name = NULL, clickable = TRUE, symbol = NULL,
  symbolSize = 4, symbolRotate = NULL, legendHoverLink = TRUE,
  polarIndex = 0, z = 2, zlevel = 0, ...)

eradar_(p, serie, name = NULL, clickable = TRUE, symbol = NULL,
  symbolSize = 4, symbolRotate = NULL, legendHoverLink = TRUE,
  polarIndex = 0, z = 2, zlevel = 0, ...)
```

Arguments

p	an echart object.
serie	value column name to plot.
name	of serie.
clickable	whether plot is clickable.
symbol	marker, see details for valid values.
symbolSize	of symbol.
symbolRotate	angle by which symbol is rotated, i.e.: 30.
legendHoverLink	enables legend hover links.
polarIndex	polar coordinates index.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
...	any other options to pass to the serie.

Details

Valid values for symbol:

- circle
- rectangle
- triangle
- diamond
- emptyCircle
- emptyRectangle
- emptyTriangle
- emptyDiamond
- heart
- droplet
- pin
- arrow
- star

See Also

official radar options docs

Examples

```
radar <- data.frame(axis = rep(LETTERS[1:6], 4), grp = sample(LETTERS[4:9], 24, replace = TRUE),
                    value = runif(24, 2, 10))

radar %>%
  group_by_("grp") %>%
  echart_("axis") %>%
  eradar_("value", symbolSize = 0) %>%
  elegend() %>%
  etheme("macarons")

radar %>%
  group_by_("grp") %>%
  echart_("axis") %>%
  eradar_("value", symbolSize = htmlwidgets::JS("function(value){return(value)}")) %>%
  elegend() %>%
  etheme("roma")
```

---

escatter *Add scatter*

Description

Add scatter serie.

Usage

```
escatter(p, serie, size = NULL, name = NULL, clickable = TRUE,
  symbol = NULL, symbolSize = 4, symbolRotate = NULL, large = FALSE,
  largeThreshold = 2000, legendHoverLink = TRUE, z = 2, zlevel = 0, ...)

escatter_(p, serie, size = NULL, name = NULL, clickable = TRUE,
  symbol = NULL, symbolSize = 4, symbolRotate = NULL, large = FALSE,
  largeThreshold = 2000, legendHoverLink = TRUE, z = 2, zlevel = 0, ...)
```

Arguments

p	an echart object.
serie	value column name to plot.
size	size of points/bubble.
name	of serie.
clickable	whether plot is clickable.

symbol	marker, see details for valid values.
symbolSize	of symbol.
symbolRotate	angle by which symbol is rotated, i.e.: 30.
large	enables large scale scatter.
largeThreshold	threshold of large scale scatter anto-switch.
legendHoverLink	enables legend hover links.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
...	any other options to pass to the serie.

Details

Valid values for symbol:

- circle
- rectangle
- triangle
- diamond
- emptyCircle
- emptyRectangle
- emptyTriangle
- emptyDiamond
- heart
- droplet
- pin
- arrow
- star

See Also

official scatter options docs

Examples

```
mtcars %>%
  echart_("disp") %>%
  escatter_("mpg", symbol = "emptyCircle") %>%
  exAxis()

mtcars %>%
  echart(disp) %>%
  escatter(mpg, qsec, symbolSize = 15) %>%
  exAxis_value(axisLabel = list(show = FALSE)) %>%
  etheme("mint") %>%
  eanimation(animationEasing = "ElasticOut")
```

`etheme` *Add theme*

Description

Add a theme.

Usage

```
etheme(p, theme = "default")
```

Arguments

`p`	an echart object.
`theme,`	see details for valid values.

Details

valid themes:

- `default`
- `mint`
- `macarons`
- `macarons2`
- `green`
- `blue`
- `dark`
- `gray`
- `helianthus`
- `red`
- `roma`
- `sakura`
- `shine`
- `infographic`
- `solarlight`

Examples

```
mtcars %>%
  echart(disp) %>%
  ebar(qsec) %>%
  ebar(mpg) %>%
  etheme("roma")
```

`etitle` *Add title*

Description

Add chart title and subtitles.

Usage

```
etitle(p, text, subtext, link, sublink, target = "blank",
  subtarget = "blank", x = "left", y = "top",
  backgroundColor = "rgba(0,0,0,0)", borderColor = "#ccc",
  borderWidth = 0, padding = 5, itemGap = 5, zlevel = 0, z = 6,
  show = TRUE, ...)
```

Arguments

`p`	an echart object.
`text`	title.
`subtext`	subtitle.
`link`	hyperlink.
`sublink`	subtext hyperlink.
`target`	`link` opening window: `self` or `blank`.
`subtarget`	`sublink` opening window: `self` or `blank`.
`x`	positon of title, `left` or `right`.
`y`	postion of title, `top`, `bottom` or `center`.
`backgroundColor`	background color.
`borderColor`	border color.
`borderWidth`	width of border.
`padding`	padding.
`itemGap`	gap between title and subtitle.
`z, zlevel`	first and second grade cascading control, the higher z the closer to the top.
`show`	whether to show the title.
`...`	any other options to pass to title

See Also

official title docs

Examples

```
mtcars %>%
  echart(disp) %>%
  eline(mpg) %>%
  etitle("MPG vs DISP", "Made with EChart", link = "http://echarts.baidu.com", target = "blank")
```

---

etoolbox *Setup toolbox*

Description

Setup toolbox

Usage

```
etoolbox(p, show = TRUE, zlevel = 0, z = 6, orient = "horizontal",
  x = "right", y = "top", backgroundColor = "rgba(0,0,0,0)",
  borderColor = "#ccc", borderWidth = 0, padding = 5, itemGap = 10,
  itemSize = 16, color = NULL, disableColor = "#ddd",
  effectiveColor = "red", showTitle = TRUE, textStyle = NULL, ...)
```

Arguments

p	an echart object.
show	whether to show the toolbox.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
orient	toolbox orientation, horizontal or vertical.
x	horizontal alignment, left, right.
y	vertical alignment, top, center, bottom.
backgroundColor	background color.
borderColor	border color.
borderWidth	border width.
padding	padding.
itemGap	space between toolbox buttons.
itemSize	size of buttons.
color	color of buttons.
disableColor	color of disabled item.
effectiveColor	color of active button.
showTitle	set to TRUE to show text.
textStyle	style of text.
...	any other options.

Examples

```
mtcars %>%
  echart(qsec) %>%
  ebar(mpg) %>%
  etoolbox() %>%
  etoolbox_magic(type = list("line", "bar"))
```

`etoolbox_feature` *Add toolbox feature*

Description

Add toolbox feature.

Usage

```
etoolbox_feature(p, mark, dataZoom, dataView, magicType, restore, saveAsImage)
```

Arguments

p	an echart object.
mark	markLine icons see etoolbox_mark.
dataZoom	dataZoom icons etoolbox_zoom.
dataView	dataView icons etoolbox_view.
magicType	magicType icons etoolbox_magic.
restore	restore icon etoolbox_restore.
saveAsImage	saveAsImage icon etoolbox_save.

Examples

```
mtcars %>%
  echart(qsec) %>%
  ebar(mpg) %>%
  etoolbox() %>%
  etoolbox_magic(type = list("line", "bar")) %>%
  etoolbox_feature(restore = list(show = TRUE))
```

etoolbox_full *Add all elements of toolbox*

Description

Adds toolbok mark, restor, save, and view.

Usage

```
etoolbox_full(p, ...)
```

Arguments

p	an echart object.
...	any other option to pass to etoolbox.

Details

Adds mark, restore, save, view and zoom buttons

etoolbox_magic *Add toolbox magic buttons*

Description

Enable changing chart type.

Usage

```
etoolbox_magic(p, show = TRUE, type = list(), title, ...)
```

Arguments

p	an echart object.
show	wehtehr to show magic buttons.
type	chart types, see details.
title	titles of charts.
...	any other options to pass to magic feature.

Details

Pass a `list` to `type`, valid values are:

- `line`
- `bar`
- `stack`
- `tiled`
- `force`
- `chord`
- `pie`
- `funnel`

Examples

```
mtcars %>%
  echart(disp) %>%
  ebar(mpg, stack = "grp") %>% # stack
  ebar(qsec, stack = "grp") %>% # stack
  ebar(wt) %>% # not stacked
  etooltip() %>%
  elegend() %>%
  etoolbox() %>%
  etoolbox_magic(type = list("bar", "line", "stack", "tiled"))
```

---

etoolbox_mark *Add toolbox feature mark button*

Description

Enable marking chart.

Usage

```
etoolbox_mark(p, show = TRUE, title = list(mark = "Mark", markUndo = "Undo",
  markClear = "Clear"), lineStyle = list(color = "#1e90ff", typed = "dashed",
  width = 2, shadowColor = "rgba(0,0,0,0)", shadowBlur = 5, shadowOffsetX = 3,
  shadowOffsetY = 3))
```

Arguments

`p`	an echart object.
`show`	whether to show mark.
`title`	mark button title.
`lineStyle`	style of marked line.

Examples

```
mtcars %>%
  echart(qsec) %>%
  ebar(mpg) %>%
  etoolbox() %>%
  etoolbox_mark()
```

---

etoolbox_restore *Add toolbox restore button*

Description

Add toolbox restore button.

Usage

```
etoolbox_restore(p, show = TRUE, title = "Reset")
```

Arguments

p	an echart object.
show	whether to show button.
title	title of button.

Examples

```
mtcars %>%
  echart(disp) %>%
  ebar(mpg, stack = "grp") %>% # stack
  ebar(qsec, stack = "grp") %>% # stack
  ebar(wt) %>% # not stacked
  etoolbox_restore()
```

---

etoolbox_save *Add toolbox save as image button*

Description

Add save as image button.

Usage

```
etoolbox_save(p, show = TRUE, title = "Save as image", type = "png",
  name = "echarts", lang = "Save")
```

Arguments

p	an echart object.
show	whether to show the button.
title	title of button.
type	image type
name	of file.
lang	text.

Examples

```
mtcars %>%
  echart(disp) %>%
  ebar(mpg, stack = "grp") %>% # stack
  ebar(qsec, stack = "grp") %>% # stack
  etoolbox() %>%
  etoolbox_save()
```

---

etoolbox_view *Add toolbox data view*

Description

Enables viewing data table.

Usage

```
etoolbox_view(p, show = TRUE, title = "View", readOnly = FALSE,
  lang = list("Data View", "close", "refresh"), ...)
```

Arguments

p	an echart object.
show	whether to show data view.
title	button title.
readOnly	set as read-only.
lang	default text.
...	any other parameters to pass to data view.

Examples

```
mtcars %>%
  echart(qsec) %>%
  ebar(mpg) %>%
  etoolbox() %>%
  etoolbox_view()
```

etoolbox_zoom *Add toolbox zoom button*

Description

Add zoom feature.

Usage

```
etoolbox_zoom(p, show = TRUE, title = list(dataZoom = "Area Zoom",
  dataZoomReset = "Reset"))
```

Arguments

p	an echart object.
show	whether to show zoom.
title	button title.

Examples

```
mtcars %>%
  echart(qsec) %>%
  ebar(mpg) %>%
  etoolbox() %>%
  etoolbox_zoom()
```

etooltip *Add tooltip*

Description

Customise tooltip.

Usage

```
etooltip(p, show = TRUE, trigger = "axis", zlevel = 1, z = 8,
  showContent = TRUE, position = NULL, formatter = NULL,
  islandFormatter = "{a} < br/>{b} : {c}", showDelay = 20,
  hideDelay = 100, transitionDuration = 0.4, enterable = FALSE,
  backgroundColor = "rgba(0,0,0,0.7)", borderColor = "#333",
  borderRadius = 4, borderWidth = 0, padding = 5, axisPointer, textStyle,
  ...)
```

Arguments

p	an echart object.
show	whether to show the tooltip.
trigger	element that triggers the tooltip, takes `item` or `axis`.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
showContent	whether to show the content of tooltip.
position	specifies position, pass a `list`, like `list(10, 10)`, fixed position; pass a function, like `htmlwidgets::JS("function([x, y]) {return [x + 10, y + 10]}")`
formatter	see http://echarts.baidu.com/echarts2/doc/option-en.html#tooltip.formatter for more details.
islandFormatter	island content formatter.
showDelay	number of milliseconds the tooltip shows.
hideDelay	number of milliseconds to wait until the tooltip is hidden when mouse out from a point or chart.
transitionDuration	duration in seconds of the animated transition.
enterable	whether to let the mouse go into the tip dom.
backgroundColor	background color.
borderColor	border color.
borderRadius	border radius.
borderWidth	border width.
padding	padding.
axisPointer	axis pointer, triggered by axis.
textStyle	tooltip text size.
...	any other options to pass to tooltip.

See Also

official tooltip docs

Examples

```
mtcars %>%
  echart(disp) %>%
  eline(mpg) %>%
  eline(qsec) %>%
  etooltip(trigger = "axis")
```

`etreemap` *Add Treemap*

Description

Add Treemap

Usage

```
etreemap(p, serie, name = NULL, itemStyle = NULL, clickable = FALSE,
  center = list("50%", "50%"), size = list("80%", "80%"), z = 2,
  zlevel = 0, ...)

etreemap_(p, serie, name = NULL, itemStyle = NULL, clickable = FALSE,
  center = list("50%", "50%"), size = list("80%", "80%"), z = 2,
  zlevel = 0, ...)
```

Arguments

`p`	an echart object.
`serie`	values to plot.
`name`	name of serie.
`itemStyle`	style of rectangles.
`clickable`	whether rectangles are clickable.
`center`	center of map.
`size`	size of chart.
`z, zlevel`	first and second grade cascading control, the higher z the closer to the top.
`...`	any other option to pass to treemap.

Examples

```
df <- data.frame(name = LETTERS[1:10], values = round(runif(10, 1, 10)))

df %>%
  echart_("name") %>%
  etreemap_("values") %>%
  etooltip(trigger = "item") %>%
  etheme("macarons")
```

evenn *Add venn*

Description

Add venn diagram

Usage

```
evenn(p, serie, name = NULL, clickable = TRUE, z = 2, zlevel = 0,
  tooltip = NULL, ...)

evenn_(p, serie, name = NULL, clickable = TRUE, z = 2, zlevel = 0,
  tooltip = NULL, ...)
```

Arguments

p	an echart object.
serie	a named vector, see details.
name	name of serie.
clickable	whether ciorcles are clickable.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
tooltip	cutomise tooltip.
...	any other argument to pass to funnel.

See Also

official venn docs

Examples

```
venn <- data.frame(name = c("banana", "pineapple", "overlap"),
  values = c(20, 50, 10))

venn %>%
  echart_("name") %>%
  evenn_("values") %>%
  etheme("macarons2")
```

ezoom *Add data zoom*

Description

Add data zoom.

Usage

```
ezoom(p, show = TRUE, zlevel = 0, z = 4, orient = "horizontal",
  backgroundColor = "rgba(0,0,0,0)", dataBackgroundColor = "#eee",
  fillerColor = "rgba(144,197,237,0.2)",
  handleColor = "rgba(70,130,180,0.8)", handleSize = 8, start = 0,
  end = 100, showDetail = TRUE, realtime = FALSE, zoomLock = FALSE, ...)
```

Arguments

p	an echart object.
show	whether to show the data zoom.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
orient	orientation, takes vertical or horinzontal.
backgroundColor	background color.
dataBackgroundColor	background color of data zoom.
fillerColor	fill color of selected area.
handleColor	color of data zoom handle.
handleSize	size of handle.
start, end	percent start and end.
showDetail	whether to show detail when dragging.
realtime	set to TRUE if using real time data.
zoomLock	when set to true, the selected area cannot be zoomed.
...	any other options to pass to data zoom.

See Also

official dataZoom docs

Examples

```
mtcars %>%
  echart(disp) %>%
  eline(mpg) %>%
  ezoom()
```

nodes *Add nodes*

Description

Add nodes for eforce.

Add nodes for eforce.

Usage

```
enodes(p, nodes, name, label, value, category, symbolSize, depth,
  ignore = FALSE, symbol = "circle", fixX = FALSE, fixY = FALSE)

enodes_(p, nodes, name, label = NULL, value = NULL, category = NULL,
  symbolSize = NULL, depth = NULL, ignore = FALSE, symbol = "circle",
  fixX = FALSE, fixY = FALSE)
```

Arguments

p	an echart object.
nodes	nodes data.frame.
name	name column.
label	nodes label column.
value	nodes value (size).
category	nodes group column.
symbolSize	nodes symbol size column.
depth	depth of nodes.
ignore	whether to ignore nodes.
symbol	nodes symbol, see details for valid values.
fixX, fixY	whether to fix x and y axis position.

Details

Valid values for symbol:

- circle
- rectangle
- triangle
- diamond
- emptyCircle
- emptyRectangle
- emptyTriangle

- emptyDiamond
- heart
- droplet
- pin
- arrow
- star

See Also

enodes eforce

Examples

```
let <- LETTERS[1:20]

edges <- data.frame(source = sample(let, 20), target = sample(let, 20),
  weight = runif(20, 5, 20))

nodes <- data.frame(name = let, value = runif(20, 5, 25), group = rep(LETTERS[1:4], 5))

echart() %>%
  eforce(itemStyle = list(normal = list(label = list(show = TRUE)))) %>% # show labels
  enodes(nodes, name, value = value, category = group) %>%
  elinks(edges, source, target)

let <- LETTERS[1:20]

edges <- data.frame(source = sample(let, 20), target = sample(let, 20),
  weight = runif(20, 5, 20))

nodes <- data.frame(name = let, value = runif(20, 5, 25), group = rep(LETTERS[1:4], 5))

echart() %>%
  eforce_(itemStyle = list(normal = list(label = list(show = TRUE)))) %>% # show labels
  enodes_(nodes, "name", value = "value", category = "group") %>%
  elinks_(edges, "source", "target")
```

---

xAxis *Customise x axis.*

Description

Customise x axis.

Customize X axis

Usage

```
exAxis(p, show = TRUE, type = "value", append = FALSE, ...)

exAxis_category(p, show = TRUE, zlevel = 0, z = 0, boundaryGap = FALSE,
  append = FALSE, ...)

exAxis_value(p, show = TRUE, min = NULL, max = NULL, zlevel = 0,
  z = 0, position = "bottom", name = NULL, nameLocation = "end",
  boundaryGap = list(0, 0), scale = FALSE, splitNumber = NULL,
  append = FALSE, ...)

exAxis_time(p, show = TRUE, zlevel = 0, z = 0, position = "bottom",
  name = NULL, nameLocation = "end", boundaryGap = list(0, 0),
  min = NULL, max = NULL, scale = FALSE, splitNumber = NULL,
  append = FALSE, ...)

exAxis_log(p, show = TRUE, zlevel = 0, z = 0, position = "bottom",
  logLabelBase = NULL, logPositive = NULL, append = FALSE, ...)
```

Arguments

p	an echart object.
show	whether to show the axis.
type	type of axis takes, `value`, `category`, `time`, `log`.
append	whether to append options to current axis or override.
...	any other parameter to pass to the axis.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
boundaryGap	whether to plot on axis line or between.
min, max	min and max values.
position	position of axis, takes `bottom`, `top`, `left` or `right`.
name	name of the axis.
nameLocation	location of `name`, takes `start` or `end`.
scale	If `FALSE`, the value axis must start with 0. If `TRUE`, you can set the minimum and maximum value of value axis as the starting and ending value of your value axis.
splitNumber	number of segments, defaults to auto split along with the min/max.
logLabelBase	base log.
logPositive	if `FALSE` negative values are not supported.

Examples

```
mtcars$models <- row.names(mtcars)

mtcars[1:10,] %>%
```

```
echart(models) %>%
eline(mpg) %>%
exAxis_category() %>%
eyAxis_value(min = 10, append = FALSE, scale = TRUE)
```

---

yAxis *Customise Y axis*

Description

Customise Y axis

Usage

```
eyAxis(p, show = TRUE, type = "value", append = FALSE, ...)

eyAxis_category(p, show = TRUE, zlevel = 0, z = 0, boundaryGap = FALSE,
  append = FALSE, ...)

eyAxis_value(p, show = TRUE, zlevel = 0, z = 0, position = "left",
  name = NULL, nameLocation = "end", nameTextStyle = list(),
  boundaryGap = list(0, 0), min = NULL, max = NULL, scale = FALSE,
  splitNumber = NULL, append = FALSE, ...)

eyAxis_time(p, show = TRUE, zlevel = 0, z = 0, position = "left",
  name = NULL, nameLocation = "end", nameTextStyle = list(),
  boundaryGap = list(0, 0), min = NULL, max = NULL, scale = FALSE,
  splitNumber = NULL, append = FALSE, ...)

eyAxis_log(p, show = TRUE, zlevel = 0, z = 0, position = "left",
  logLabelBase = NULL, logPositive = NULL, append = FALSE, ...)
```

Arguments

p	an echart object.
show	whether to show the axis.
type	type of axis takes, value, category, time, log.
append	whether to append options to current axis or override.
...	any other parameter to pass to the axis.
z, zlevel	first and second grade cascading control, the higher z the closer to the top.
boundaryGap	whether to plot on axis line or between.
position	position of axis, takes bottom, top, left or right.
name	name of the axis.
nameLocation	location of name, takes start or end.

nameTextStyle	style name text.
min, max	min and max values.
scale	If FALSE, the value axis must start with 0. If TRUE, you can set the minimum and maximum value of value axis as the starting and ending value of your value axis.
splitNumber	number of segments, defaults to auto split along with the min/max.
logLabelBase	base log.
logPositive	if FALSE negative values are not supported.

Examples

```
df <- data.frame(x = c("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"),
  y = runif(7, 1, 5))

df %>%
  echart(x) %>%
  eline(y) %>%
  exAxis_category(boundaryGap = FALSE)

df %>%
  echart(x) %>%
  ebar(y) %>%
  eyAxis_log()
```

Index