## **Python Figure Reference: layout**

* [title](https://plotly.com/python/reference/layout/#layout-title)*Code:* fig.update\_layout(title=dict(...))  
  Type: dict containing one or more of the keys listed below.
  + [font](https://plotly.com/python/reference/layout/#layout-title-font)*Code:* fig.update\_layout(title\_font=dict(...))  
    Type: dict containing one or more of the keys listed below.
  + Sets the title font. Note that the title's font used to be customized by the now deprecated `titlefont` attribute.
    - [color](https://plotly.com/python/reference/layout/#layout-title-font-color)*Code:* fig.update\_layout(title\_font\_color=<VALUE>)  
      Type: color
    - [family](https://plotly.com/python/reference/layout/#layout-title-font-family)*Code:* fig.update\_layout(title\_font\_family=<VALUE>)  
      Type: string
    - HTML font family - the typeface that will be applied by the web browser. The web browser will only be able to apply a font if it is available on the system which it operates. Provide multiple font families, separated by commas, to indicate the preference in which to apply fonts if they aren't available on the system. The Chart Studio Cloud (at <https://chart-studio.plotly.com> or on-premise) generates images on a server, where only a select number of fonts are installed and supported. These include "Arial", "Balto", "Courier New", "Droid Sans",, "Droid Serif", "Droid Sans Mono", "Gravitas One", "Old Standard TT", "Open Sans", "Overpass", "PT Sans Narrow", "Raleway", "Times New Roman".
    - [size](https://plotly.com/python/reference/layout/#layout-title-font-size)*Code:* fig.update\_layout(title\_font\_size=<VALUE>)  
      Type: number greater than or equal to 1
  + [pad](https://plotly.com/python/reference/layout/#layout-title-pad)*Code:* fig.update\_layout(title\_pad=dict(...))  
    Type: dict containing one or more of the keys listed below.
  + Sets the padding of the title. Each padding value only applies when the corresponding `xanchor`/`yanchor` value is set accordingly. E.g. for left padding to take effect, `xanchor` must be set to "left". The same rule applies if `xanchor`/`yanchor` is determined automatically. Padding is muted if the respective anchor value is "middle"/"center".
    - [b](https://plotly.com/python/reference/layout/#layout-title-pad-b)*Code:* fig.update\_layout(title\_pad\_b=<VALUE>)  
      Type: number  
      Default: 0
    - The amount of padding (in px) along the bottom of the component.
    - [l](https://plotly.com/python/reference/layout/#layout-title-pad-l)*Code:* fig.update\_layout(title\_pad\_l=<VALUE>)  
      Type: number  
      Default: 0
    - The amount of padding (in px) on the left side of the component.
    - [r](https://plotly.com/python/reference/layout/#layout-title-pad-r)*Code:* fig.update\_layout(title\_pad\_r=<VALUE>)  
      Type: number  
      Default: 0
    - The amount of padding (in px) on the right side of the component.
    - [t](https://plotly.com/python/reference/layout/#layout-title-pad-t)*Code:* fig.update\_layout(title\_pad\_t=<VALUE>)  
      Type: number  
      Default: 0
    - The amount of padding (in px) along the top of the component.
  + [text](https://plotly.com/python/reference/layout/#layout-title-text)*Code:* fig.update\_layout(title\_text=<VALUE>)  
    Type: string
  + Sets the plot's title. Note that before the existence of `title.text`, the title's contents used to be defined as the `title` attribute itself. This behavior has been deprecated.
  + [x](https://plotly.com/python/reference/layout/#layout-title-x)*Code:* fig.update\_layout(title\_x=<VALUE>)  
    Type: number between or equal to 0 and 1  
    Default: 0.5
  + Sets the x position with respect to `xref` in normalized coordinates from "0" (left) to "1" (right).
  + [xanchor](https://plotly.com/python/reference/layout/#layout-title-xanchor)*Code:* fig.update\_layout(title\_xanchor=<VALUE>)  
    Type: enumerated , one of ( "auto" | "left" | "center" | "right" )  
    Default: "auto"
  + Sets the title's horizontal alignment with respect to its x position. "left" means that the title starts at x, "right" means that the title ends at x and "center" means that the title's center is at x. "auto" divides `xref` by three and calculates the `xanchor` value automatically based on the value of `x`.
  + [xref](https://plotly.com/python/reference/layout/#layout-title-xref)*Code:* fig.update\_layout(title\_xref=<VALUE>)  
    Type: enumerated , one of ( "container" | "paper" )  
    Default: "container"
  + Sets the container `x` refers to. "container" spans the entire `width` of the plot. "paper" refers to the width of the plotting area only.
  + [y](https://plotly.com/python/reference/layout/#layout-title-y)*Code:* fig.update\_layout(title\_y=<VALUE>)  
    Type: number between or equal to 0 and 1  
    Default: "auto"
  + Sets the y position with respect to `yref` in normalized coordinates from "0" (bottom) to "1" (top). "auto" places the baseline of the title onto the vertical center of the top margin.
  + [yanchor](https://plotly.com/python/reference/layout/#layout-title-yanchor)*Code:* fig.update\_layout(title\_yanchor=<VALUE>)  
    Type: enumerated , one of ( "auto" | "top" | "middle" | "bottom" )  
    Default: "auto"
  + Sets the title's vertical alignment with respect to its y position. "top" means that the title's cap line is at y, "bottom" means that the title's baseline is at y and "middle" means that the title's midline is at y. "auto" divides `yref` by three and calculates the `yanchor` value automatically based on the value of `y`.
  + [yref](https://plotly.com/python/reference/layout/#layout-title-yref)*Code:* fig.update\_layout(title\_yref=<VALUE>)  
    Type: enumerated , one of ( "container" | "paper" )  
    Default: "container"
  + Sets the container `y` refers to. "container" spans the entire `height` of the plot. "paper" refers to the height of the plotting area only.
* [showlegend](https://plotly.com/python/reference/layout/#layout-showlegend)*Code:* fig.update\_layout(showlegend=<VALUE>)  
  Type: boolean
* Determines whether or not a legend is drawn. Default is `True` if there is a trace to show and any of these: a) Two or more traces would by default be shown in the legend. b) One pie trace is shown in the legend. c) One trace is explicitly given with `showlegend: True`.
* [legend](https://plotly.com/python/reference/layout/#layout-legend)*Code:* fig.update\_layout(legend=dict(...))  
  Type: dict containing one or more of the keys listed below.
  + [bgcolor](https://plotly.com/python/reference/layout/#layout-legend-bgcolor)*Code:* fig.update\_layout(legend\_bgcolor=<VALUE>)  
    Type: color
  + Sets the legend background color. Defaults to `layout.paper\_bgcolor`.
  + [bordercolor](https://plotly.com/python/reference/layout/#layout-legend-bordercolor)*Code:* fig.update\_layout(legend\_bordercolor=<VALUE>)  
    Type: color  
    Default: "#444"
  + Sets the color of the border enclosing the legend.
  + [borderwidth](https://plotly.com/python/reference/layout/#layout-legend-borderwidth)*Code:* fig.update\_layout(legend\_borderwidth=<VALUE>)  
    Type: number greater than or equal to 0  
    Default: 0
  + Sets the width (in px) of the border enclosing the legend.
  + [entrywidth](https://plotly.com/python/reference/layout/#layout-legend-entrywidth)*Code:* fig.update\_layout(legend\_entrywidth=<VALUE>)  
    Type: number greater than or equal to 0
  + Sets the width (in px or fraction) of the legend. Use 0 to size the entry based on the text width, when `entrywidthmode` is set to "pixels".
  + [entrywidthmode](https://plotly.com/python/reference/layout/#layout-legend-entrywidthmode)*Code:* fig.update\_layout(legend\_entrywidthmode=<VALUE>)  
    Type: enumerated , one of ( "fraction" | "pixels" )  
    Default: "pixels"
  + Determines what entrywidth means.
  + [font](https://plotly.com/python/reference/layout/#layout-legend-font)*Code:* fig.update\_layout(legend\_font=dict(...))  
    Type: dict containing one or more of the keys listed below.
  + Sets the font used to text the legend items.
    - [color](https://plotly.com/python/reference/layout/#layout-legend-font-color)*Code:* fig.update\_layout(legend\_font\_color=<VALUE>)  
      Type: color
    - [family](https://plotly.com/python/reference/layout/#layout-legend-font-family)*Code:* fig.update\_layout(legend\_font\_family=<VALUE>)  
      Type: string
    - HTML font family - the typeface that will be applied by the web browser. The web browser will only be able to apply a font if it is available on the system which it operates. Provide multiple font families, separated by commas, to indicate the preference in which to apply fonts if they aren't available on the system. The Chart Studio Cloud (at <https://chart-studio.plotly.com> or on-premise) generates images on a server, where only a select number of fonts are installed and supported. These include "Arial", "Balto", "Courier New", "Droid Sans",, "Droid Serif", "Droid Sans Mono", "Gravitas One", "Old Standard TT", "Open Sans", "Overpass", "PT Sans Narrow", "Raleway", "Times New Roman".
    - [size](https://plotly.com/python/reference/layout/#layout-legend-font-size)*Code:* fig.update\_layout(legend\_font\_size=<VALUE>)  
      Type: number greater than or equal to 1
  + [groupclick](https://plotly.com/python/reference/layout/#layout-legend-groupclick)*Code:* fig.update\_layout(legend\_groupclick=<VALUE>)  
    Type: enumerated , one of ( "toggleitem" | "togglegroup" )  
    Default: "togglegroup"
  + Determines the behavior on legend group item click. "toggleitem" toggles the visibility of the individual item clicked on the graph. "togglegroup" toggles the visibility of all items in the same legendgroup as the item clicked on the graph.
  + [grouptitlefont](https://plotly.com/python/reference/layout/#layout-legend-grouptitlefont)*Code:* fig.update\_layout(legend\_grouptitlefont=dict(...))  
    Type: dict containing one or more of the keys listed below.
  + Sets the font for group titles in legend. Defaults to `legend.font` with its size increased about 10%.
    - [color](https://plotly.com/python/reference/layout/#layout-legend-grouptitlefont-color)*Code:* fig.update\_layout(legend\_grouptitlefont\_color=<VALUE>)  
      Type: color
    - [family](https://plotly.com/python/reference/layout/#layout-legend-grouptitlefont-family)*Code:* fig.update\_layout(legend\_grouptitlefont\_family=<VALUE>)  
      Type: string
    - HTML font family - the typeface that will be applied by the web browser. The web browser will only be able to apply a font if it is available on the system which it operates. Provide multiple font families, separated by commas, to indicate the preference in which to apply fonts if they aren't available on the system. The Chart Studio Cloud (at <https://chart-studio.plotly.com> or on-premise) generates images on a server, where only a select number of fonts are installed and supported. These include "Arial", "Balto", "Courier New", "Droid Sans",, "Droid Serif", "Droid Sans Mono", "Gravitas One", "Old Standard TT", "Open Sans", "Overpass", "PT Sans Narrow", "Raleway", "Times New Roman".
    - [size](https://plotly.com/python/reference/layout/#layout-legend-grouptitlefont-size)*Code:* fig.update\_layout(legend\_grouptitlefont\_size=<VALUE>)  
      Type: number greater than or equal to 1
  + [itemclick](https://plotly.com/python/reference/layout/#layout-legend-itemclick)*Code:* fig.update\_layout(legend\_itemclick=<VALUE>)  
    Type: enumerated , one of ( "toggle" | "toggleothers" | False )  
    Default: "toggle"
  + Determines the behavior on legend item click. "toggle" toggles the visibility of the item clicked on the graph. "toggleothers" makes the clicked item the sole visible item on the graph. "False" disables legend item click interactions.
  + [itemdoubleclick](https://plotly.com/python/reference/layout/#layout-legend-itemdoubleclick)*Code:* fig.update\_layout(legend\_itemdoubleclick=<VALUE>)  
    Type: enumerated , one of ( "toggle" | "toggleothers" | False )  
    Default: "toggleothers"
  + Determines the behavior on legend item double-click. "toggle" toggles the visibility of the item clicked on the graph. "toggleothers" makes the clicked item the sole visible item on the graph. "False" disables legend item double-click interactions.
  + [itemsizing](https://plotly.com/python/reference/layout/#layout-legend-itemsizing)*Code:* fig.update\_layout(legend\_itemsizing=<VALUE>)  
    Type: enumerated , one of ( "trace" | "constant" )  
    Default: "trace"
  + Determines if the legend items symbols scale with their corresponding "trace" attributes or remain "constant" independent of the symbol size on the graph.
  + [itemwidth](https://plotly.com/python/reference/layout/#layout-legend-itemwidth)*Code:* fig.update\_layout(legend\_itemwidth=<VALUE>)  
    Type: number greater than or equal to 30  
    Default: 30
  + Sets the width (in px) of the legend item symbols (the part other than the title.text).
  + [orientation](https://plotly.com/python/reference/layout/#layout-legend-orientation)*Code:* fig.update\_layout(legend\_orientation=<VALUE>)  
    Type: enumerated , one of ( "v" | "h" )  
    Default: "v"
  + Sets the orientation of the legend.
  + [title](https://plotly.com/python/reference/layout/#layout-legend-title)*Code:* fig.update\_layout(legend\_title=dict(...))  
    Type: dict containing one or more of the keys listed below.
    - [font](https://plotly.com/python/reference/layout/#layout-legend-title-font)*Code:* fig.update\_layout(legend\_title\_font=dict(...))  
      Type: dict containing one or more of the keys listed below.
    - Sets this legend's title font. Defaults to `legend.font` with its size increased about 20%.
      * [color](https://plotly.com/python/reference/layout/#layout-legend-title-font-color)*Code:* fig.update\_layout(legend\_title\_font\_color=<VALUE>)  
        Type: color
      * [family](https://plotly.com/python/reference/layout/#layout-legend-title-font-family)*Code:* fig.update\_layout(legend\_title\_font\_family=<VALUE>)  
        Type: string
      * HTML font family - the typeface that will be applied by the web browser. The web browser will only be able to apply a font if it is available on the system which it operates. Provide multiple font families, separated by commas, to indicate the preference in which to apply fonts if they aren't available on the system. The Chart Studio Cloud (at <https://chart-studio.plotly.com> or on-premise) generates images on a server, where only a select number of fonts are installed and supported. These include "Arial", "Balto", "Courier New", "Droid Sans",, "Droid Serif", "Droid Sans Mono", "Gravitas One", "Old Standard TT", "Open Sans", "Overpass", "PT Sans Narrow", "Raleway", "Times New Roman".
      * [size](https://plotly.com/python/reference/layout/#layout-legend-title-font-size)*Code:* fig.update\_layout(legend\_title\_font\_size=<VALUE>)  
        Type: number greater than or equal to 1
    - [side](https://plotly.com/python/reference/layout/#layout-legend-title-side)*Code:* fig.update\_layout(legend\_title\_side=<VALUE>)  
      Type: enumerated , one of ( "top" | "left" | "top left" )
    - Determines the location of legend's title with respect to the legend items. Defaulted to "top" with `orientation` is "h". Defaulted to "left" with `orientation` is "v". The "top left" options could be used to expand legend area in both x and y sides.
    - [text](https://plotly.com/python/reference/layout/#layout-legend-title-text)*Code:* fig.update\_layout(legend\_title\_text=<VALUE>)  
      Type: string  
      Default: ""
    - Sets the title of the legend.
  + [tracegroupgap](https://plotly.com/python/reference/layout/#layout-legend-tracegroupgap)*Code:* fig.update\_layout(legend\_tracegroupgap=<VALUE>)  
    Type: number greater than or equal to 0  
    Default: 10
  + Sets the amount of vertical space (in px) between legend groups.
  + [traceorder](https://plotly.com/python/reference/layout/#layout-legend-traceorder)*Code:* fig.update\_layout(legend\_traceorder=<VALUE>)  
    Type: flaglist string. Any combination of "reversed", "grouped" joined with a "+" OR "normal".  
    Examples: "reversed", "grouped", "reversed+grouped", "normal"
  + Determines the order at which the legend items are displayed. If "normal", the items are displayed top-to-bottom in the same order as the input data. If "reversed", the items are displayed in the opposite order as "normal". If "grouped", the items are displayed in groups (when a trace `legendgroup` is provided). if "grouped+reversed", the items are displayed in the opposite order as "grouped".
  + [uirevision](https://plotly.com/python/reference/layout/#layout-legend-uirevision)*Code:* fig.update\_layout(legend\_uirevision=<VALUE>)  
    Type: number or categorical coordinate string
  + Controls persistence of legend-driven changes in trace and pie label visibility. Defaults to `layout.uirevision`.
  + [valign](https://plotly.com/python/reference/layout/#layout-legend-valign)*Code:* fig.update\_layout(legend\_valign=<VALUE>)  
    Type: enumerated , one of ( "top" | "middle" | "bottom" )  
    Default: "middle"
  + Sets the vertical alignment of the symbols with respect to their associated text.
  + [x](https://plotly.com/python/reference/layout/#layout-legend-x)*Code:* fig.update\_layout(legend\_x=<VALUE>)  
    Type: number between or equal to -2 and 3
  + Sets the x position (in normalized coordinates) of the legend. Defaults to "1.02" for vertical legends and defaults to "0" for horizontal legends.
  + [xanchor](https://plotly.com/python/reference/layout/#layout-legend-xanchor)*Code:* fig.update\_layout(legend\_xanchor=<VALUE>)  
    Type: enumerated , one of ( "auto" | "left" | "center" | "right" )  
    Default: "left"
  + Sets the legend's horizontal position anchor. This anchor binds the `x` position to the "left", "center" or "right" of the legend. Value "auto" anchors legends to the right for `x` values greater than or equal to 2/3, anchors legends to the left for `x` values less than or equal to 1/3 and anchors legends with respect to their center otherwise.
  + [y](https://plotly.com/python/reference/layout/#layout-legend-y)*Code:* fig.update\_layout(legend\_y=<VALUE>)  
    Type: number between or equal to -2 and 3
  + Sets the y position (in normalized coordinates) of the legend. Defaults to "1" for vertical legends, defaults to "-0.1" for horizontal legends on graphs w/o range sliders and defaults to "1.1" for horizontal legends on graph with one or multiple range sliders.
  + [yanchor](https://plotly.com/python/reference/layout/#layout-legend-yanchor)*Code:* fig.update\_layout(legend\_yanchor=<VALUE>)  
    Type: enumerated , one of ( "auto" | "top" | "middle" | "bottom" )
  + Sets the legend's vertical position anchor This anchor binds the `y` position to the "top", "middle" or "bottom" of the legend. Value "auto" anchors legends at their bottom for `y` values less than or equal to 1/3, anchors legends to at their top for `y` values greater than or equal to 2/3 and anchors legends with respect to their middle otherwise.
* [margin](https://plotly.com/python/reference/layout/#layout-margin)*Code:* fig.update\_layout(margin=dict(...))  
  Type: dict containing one or more of the keys listed below.
  + [autoexpand](https://plotly.com/python/reference/layout/#layout-margin-autoexpand)*Code:* fig.update\_layout(margin\_autoexpand=<VALUE>)  
    Type: boolean  
    Default: True
  + Turns on/off margin expansion computations. Legends, colorbars, updatemenus, sliders, axis rangeselector and rangeslider are allowed to push the margins by defaults.
  + [b](https://plotly.com/python/reference/layout/#layout-margin-b)*Code:* fig.update\_layout(margin\_b=<VALUE>)  
    Type: number greater than or equal to 0  
    Default: 80
  + Sets the bottom margin (in px).
  + [l](https://plotly.com/python/reference/layout/#layout-margin-l)*Code:* fig.update\_layout(margin\_l=<VALUE>)  
    Type: number greater than or equal to 0  
    Default: 80
  + Sets the left margin (in px).
  + [pad](https://plotly.com/python/reference/layout/#layout-margin-pad)*Code:* fig.update\_layout(margin\_pad=<VALUE>)  
    Type: number greater than or equal to 0  
    Default: 0
  + Sets the amount of padding (in px) between the plotting area and the axis lines
  + [r](https://plotly.com/python/reference/layout/#layout-margin-r)*Code:* fig.update\_layout(margin\_r=<VALUE>)  
    Type: number greater than or equal to 0  
    Default: 80
  + Sets the right margin (in px).
  + [t](https://plotly.com/python/reference/layout/#layout-margin-t)*Code:* fig.update\_layout(margin\_t=<VALUE>)  
    Type: number greater than or equal to 0  
    Default: 100
  + Sets the top margin (in px).
* [autosize](https://plotly.com/python/reference/layout/#layout-autosize)*Code:* fig.update\_layout(autosize=<VALUE>)  
  Type: boolean
* Determines whether or not a layout width or height that has been left undefined by the user is initialized on each relayout. Note that, regardless of this attribute, an undefined layout width or height is always initialized on the first call to plot.
* [width](https://plotly.com/python/reference/layout/#layout-width)*Code:* fig.update\_layout(width=<VALUE>)  
  Type: number greater than or equal to 10  
  Default: 700
* Sets the plot's width (in px).
* [height](https://plotly.com/python/reference/layout/#layout-height)*Code:* fig.update\_layout(height=<VALUE>)  
  Type: number greater than or equal to 10  
  Default: 450
* Sets the plot's height (in px).
* [font](https://plotly.com/python/reference/layout/#layout-font)*Code:* fig.update\_layout(font=dict(...))  
  Type: dict containing one or more of the keys listed below.
* Sets the global font. Note that fonts used in traces and other layout components inherit from the global font.
  + [color](https://plotly.com/python/reference/layout/#layout-font-color)*Code:* fig.update\_layout(font\_color=<VALUE>)  
    Type: color  
    Default: "#444"
  + [family](https://plotly.com/python/reference/layout/#layout-font-family)*Code:* fig.update\_layout(font\_family=<VALUE>)  
    Type: string  
    Default: ""Open Sans", verdana, arial, sans-serif"
  + HTML font family - the typeface that will be applied by the web browser. The web browser will only be able to apply a font if it is available on the system which it operates. Provide multiple font families, separated by commas, to indicate the preference in which to apply fonts if they aren't available on the system. The Chart Studio Cloud (at <https://chart-studio.plotly.com> or on-premise) generates images on a server, where only a select number of fonts are installed and supported. These include "Arial", "Balto", "Courier New", "Droid Sans",, "Droid Serif", "Droid Sans Mono", "Gravitas One", "Old Standard TT", "Open Sans", "Overpass", "PT Sans Narrow", "Raleway", "Times New Roman".
  + [size](https://plotly.com/python/reference/layout/#layout-font-size)*Code:* fig.update\_layout(font\_size=<VALUE>)  
    Type: number greater than or equal to 1  
    Default: 12
* [uniformtext](https://plotly.com/python/reference/layout/#layout-uniformtext)*Code:* fig.update\_layout(uniformtext=dict(...))  
  Type: dict containing one or more of the keys listed below.
  + [minsize](https://plotly.com/python/reference/layout/#layout-uniformtext-minsize)*Code:* fig.update\_layout(uniformtext\_minsize=<VALUE>)  
    Type: number greater than or equal to 0  
    Default: 0
  + Sets the minimum text size between traces of the same type.
  + [mode](https://plotly.com/python/reference/layout/#layout-uniformtext-mode)*Code:* fig.update\_layout(uniformtext\_mode=<VALUE>)  
    Type: enumerated , one of ( False | "hide" | "show" )
  + Determines how the font size for various text elements are uniformed between each trace type. If the computed text sizes were smaller than the minimum size defined by `uniformtext.minsize` using "hide" option hides the text; and using "show" option shows the text without further downscaling. Please note that if the size defined by `minsize` is greater than the font size defined by trace, then the `minsize` is used.
* [separators](https://plotly.com/python/reference/layout/#layout-separators)*Code:* fig.update\_layout(separators=<VALUE>)  
  Type: string
* Sets the decimal and thousand separators. For example, ". " puts a '.' before decimals and a space between thousands. In English locales, dflt is ".," but other locales may alter this default.
* [paper\_bgcolor](https://plotly.com/python/reference/layout/#layout-paper_bgcolor)*Code:* fig.update\_layout(paper\_bgcolor=<VALUE>)  
  Type: color  
  Default: "#fff"
* Sets the background color of the paper where the graph is drawn.
* [plot\_bgcolor](https://plotly.com/python/reference/layout/#layout-plot_bgcolor)*Code:* fig.update\_layout(plot\_bgcolor=<VALUE>)  
  Type: color  
  Default: "#fff"
* Sets the background color of the plotting area in-between x and y axes.
* [autotypenumbers](https://plotly.com/python/reference/layout/#layout-autotypenumbers)*Code:* fig.update\_layout(autotypenumbers=<VALUE>)  
  Type: enumerated , one of ( "convert types" | "strict" )  
  Default: "convert types"
* Using "strict" a numeric string in trace data is not converted to a number. Using "convert types" a numeric string in trace data may be treated as a number during automatic axis `type` detection. This is the default value; however it could be overridden for individual axes.
* [colorscale](https://plotly.com/python/reference/layout/#layout-colorscale)*Code:* fig.update\_layout(colorscale=dict(...))  
  Type: dict containing one or more of the keys listed below.
  + [diverging](https://plotly.com/python/reference/layout/#layout-colorscale-diverging)*Code:* fig.update\_layout(colorscale\_diverging=<VALUE>)  
    Type: colorscale  
    Default: [[0, rgb(5,10,172)], [0.35, rgb(106,137,247)], [0.5, rgb(190,190,190)], [0.6, rgb(220,170,132)], [0.7, rgb(230,145,90)], [1, rgb(178,10,28)], ]
  + Sets the default diverging colorscale. Note that `autocolorscale` must be True for this attribute to work.
  + [sequential](https://plotly.com/python/reference/layout/#layout-colorscale-sequential)*Code:* fig.update\_layout(colorscale\_sequential=<VALUE>)  
    Type: colorscale  
    Default: [[0, rgb(220,220,220)], [0.2, rgb(245,195,157)], [0.4, rgb(245,160,105)], [1, rgb(178,10,28)], ]
  + Sets the default sequential colorscale for positive values. Note that `autocolorscale` must be True for this attribute to work.
  + [sequentialminus](https://plotly.com/python/reference/layout/#layout-colorscale-sequentialminus)*Code:* fig.update\_layout(colorscale\_sequentialminus=<VALUE>)  
    Type: colorscale  
    Default: [[0, rgb(5,10,172)], [0.35, rgb(40,60,190)], [0.5, rgb(70,100,245)], [0.6, rgb(90,120,245)], [0.7, rgb(106,137,247)], [1, rgb(220,220,220)], ]
  + Sets the default sequential colorscale for negative values. Note that `autocolorscale` must be True for this attribute to work.
* [colorway](https://plotly.com/python/reference/layout/#layout-colorway)*Code:* fig.update\_layout(colorway=<VALUE>)  
  Type: colorlist  
  Default: [#1f77b4, #ff7f0e, #2ca02c, #d62728, #9467bd, #8c564b, #e377c2, #7f7f7f, #bcbd22, #17becf]
* Sets the default trace colors.
* [modebar](https://plotly.com/python/reference/layout/#layout-modebar)*Code:* fig.update\_layout(modebar=dict(...))  
  Type: dict containing one or more of the keys listed below.
  + [activecolor](https://plotly.com/python/reference/layout/#layout-modebar-activecolor)*Code:* fig.update\_layout(modebar\_activecolor=<VALUE>)  
    Type: color
  + Sets the color of the active or hovered on icons in the modebar.
  + [add](https://plotly.com/python/reference/layout/#layout-modebar-add)*Code:* fig.update\_layout(modebar\_add=<VALUE>)  
    Type: string or array of strings  
    Default: ""
  + Determines which predefined modebar buttons to add. Please note that these buttons will only be shown if they are compatible with all trace types used in a graph. Similar to `config.modeBarButtonsToAdd` option. This may include "v1hovermode", "hoverclosest", "hovercompare", "togglehover", "togglespikelines", "drawline", "drawopenpath", "drawclosedpath", "drawcircle", "drawrect", "eraseshape".
  + [bgcolor](https://plotly.com/python/reference/layout/#layout-modebar-bgcolor)*Code:* fig.update\_layout(modebar\_bgcolor=<VALUE>)  
    Type: color
  + Sets the background color of the modebar.
  + [color](https://plotly.com/python/reference/layout/#layout-modebar-color)*Code:* fig.update\_layout(modebar\_color=<VALUE>)  
    Type: color
  + Sets the color of the icons in the modebar.
  + [orientation](https://plotly.com/python/reference/layout/#layout-modebar-orientation)*Code:* fig.update\_layout(modebar\_orientation=<VALUE>)  
    Type: enumerated , one of ( "v" | "h" )  
    Default: "h"
  + Sets the orientation of the modebar.
  + [remove](https://plotly.com/python/reference/layout/#layout-modebar-remove)*Code:* fig.update\_layout(modebar\_remove=<VALUE>)  
    Type: string or array of strings  
    Default: ""
  + Determines which predefined modebar buttons to remove. Similar to `config.modeBarButtonsToRemove` option. This may include "autoScale2d", "autoscale", "editInChartStudio", "editinchartstudio", "hoverCompareCartesian", "hovercompare", "lasso", "lasso2d", "orbitRotation", "orbitrotation", "pan", "pan2d", "pan3d", "reset", "resetCameraDefault3d", "resetCameraLastSave3d", "resetGeo", "resetSankeyGroup", "resetScale2d", "resetViewMapbox", "resetViews", "resetcameradefault", "resetcameralastsave", "resetsankeygroup", "resetscale", "resetview", "resetviews", "select", "select2d", "sendDataToCloud", "senddatatocloud", "tableRotation", "tablerotation", "toImage", "toggleHover", "toggleSpikelines", "togglehover", "togglespikelines", "toimage", "zoom", "zoom2d", "zoom3d", "zoomIn2d", "zoomInGeo", "zoomInMapbox", "zoomOut2d", "zoomOutGeo", "zoomOutMapbox", "zoomin", "zoomout".
  + [uirevision](https://plotly.com/python/reference/layout/#layout-modebar-uirevision)*Code:* fig.update\_layout(modebar\_uirevision=<VALUE>)  
    Type: number or categorical coordinate string
  + Controls persistence of user-driven changes related to the modebar, including `hovermode`, `dragmode`, and `showspikes` at both the root level and inside subplots. Defaults to `layout.uirevision`.
* [hovermode](https://plotly.com/python/reference/layout/#layout-hovermode)*Code:* fig.update\_layout(hovermode=<VALUE>)  
  Type: enumerated , one of ( "x" | "y" | "closest" | False | "x unified" | "y unified" )  
  Default: "closest"
* Determines the mode of hover interactions. If "closest", a single hoverlabel will appear for the "closest" point within the `hoverdistance`. If "x" (or "y"), multiple hoverlabels will appear for multiple points at the "closest" x- (or y-) coordinate within the `hoverdistance`, with the caveat that no more than one hoverlabel will appear per trace. If "x unified" (or "y unified"), a single hoverlabel will appear multiple points at the closest x- (or y-) coordinate within the `hoverdistance` with the caveat that no more than one hoverlabel will appear per trace. In this mode, spikelines are enabled by default perpendicular to the specified axis. If False, hover interactions are disabled.
* [clickmode](https://plotly.com/python/reference/layout/#layout-clickmode)*Code:* fig.update\_layout(clickmode=<VALUE>)  
  Type: flaglist string. Any combination of "event", "select" joined with a "+" OR "none".  
  Examples: "event", "select", "event+select", "none"  
  Default: "event"
* Determines the mode of single click interactions. "event" is the default value and emits the `plotly\_click` event. In addition this mode emits the `plotly\_selected` event in drag modes "lasso" and "select", but with no event data attached (kept for compatibility reasons). The "select" flag enables selecting single data points via click. This mode also supports persistent selections, meaning that pressing Shift while clicking, adds to / subtracts from an existing selection. "select" with `hovermode`: "x" can be confusing, consider explicitly setting `hovermode`: "closest" when using this feature. Selection events are sent accordingly as long as "event" flag is set as well. When the "event" flag is missing, `plotly\_click` and `plotly\_selected` events are not fired.
* [dragmode](https://plotly.com/python/reference/layout/#layout-dragmode)*Code:* fig.update\_layout(dragmode=<VALUE>)  
  Type: enumerated , one of ( "zoom" | "pan" | "select" | "lasso" | "drawclosedpath" | "drawopenpath" | "drawline" | "drawrect" | "drawcircle" | "orbit" | "turntable" | False )  
  Default: "zoom"
* Determines the mode of drag interactions. "select" and "lasso" apply only to scatter traces with markers or text. "orbit" and "turntable" apply only to 3D scenes.
* [selectdirection](https://plotly.com/python/reference/layout/#layout-selectdirection)*Code:* fig.update\_layout(selectdirection=<VALUE>)  
  Type: enumerated , one of ( "h" | "v" | "d" | "any" )  
  Default: "any"
* When `dragmode` is set to "select", this limits the selection of the drag to horizontal, vertical or diagonal. "h" only allows horizontal selection, "v" only vertical, "d" only diagonal and "any" sets no limit.
* [activeselection](https://plotly.com/python/reference/layout/#layout-activeselection)*Code:* fig.update\_layout(activeselection=dict(...))  
  Type: dict containing one or more of the keys listed below.
  + [fillcolor](https://plotly.com/python/reference/layout/#layout-activeselection-fillcolor)*Code:* fig.update\_layout(activeselection\_fillcolor=<VALUE>)  
    Type: color  
    Default: "rgba(0,0,0,0)"
  + Sets the color filling the active selection' interior.
  + [opacity](https://plotly.com/python/reference/layout/#layout-activeselection-opacity)*Code:* fig.update\_layout(activeselection\_opacity=<VALUE>)  
    Type: number between or equal to 0 and 1  
    Default: 0.5
  + Sets the opacity of the active selection.
* [newselection](https://plotly.com/python/reference/layout/#layout-newselection)*Code:* fig.update\_layout(newselection=dict(...))  
  Type: dict containing one or more of the keys listed below.
  + [line](https://plotly.com/python/reference/layout/#layout-newselection-line)*Code:* fig.update\_layout(newselection\_line=dict(...))  
    Type: dict containing one or more of the keys listed below.
    - [color](https://plotly.com/python/reference/layout/#layout-newselection-line-color)*Code:* fig.update\_layout(newselection\_line\_color=<VALUE>)  
      Type: color
    - Sets the line color. By default uses either dark grey or white to increase contrast with background color.
    - [dash](https://plotly.com/python/reference/layout/#layout-newselection-line-dash)*Code:* fig.update\_layout(newselection\_line\_dash=<VALUE>)  
      Type: string  
      Default: "dot"
    - Sets the dash style of lines. Set to a dash type string ("solid", "dot", "dash", "longdash", "dashdot", or "longdashdot") or a dash length list in px (eg "5px,10px,2px,2px").
    - [width](https://plotly.com/python/reference/layout/#layout-newselection-line-width)*Code:* fig.update\_layout(newselection\_line\_width=<VALUE>)  
      Type: number greater than or equal to 1  
      Default: 1
    - Sets the line width (in px).
  + [mode](https://plotly.com/python/reference/layout/#layout-newselection-mode)*Code:* fig.update\_layout(newselection\_mode=<VALUE>)  
    Type: enumerated , one of ( "immediate" | "gradual" )  
    Default: "immediate"
  + Describes how a new selection is created. If `immediate`, a new selection is created after first mouse up. If `gradual`, a new selection is not created after first mouse. By adding to and subtracting from the initial selection, this option allows declaring extra outlines of the selection.
* [hoverdistance](https://plotly.com/python/reference/layout/#layout-hoverdistance)*Code:* fig.update\_layout(hoverdistance=<VALUE>)  
  Type: integer greater than or equal to -1  
  Default: 20
* Sets the default distance (in pixels) to look for data to add hover labels (-1 means no cutoff, 0 means no looking for data). This is only a real distance for hovering on point-like objects, like scatter points. For area-like objects (bars, scatter fills, etc) hovering is on inside the area and off outside, but these objects will not supersede hover on point-like objects in case of conflict.
* [spikedistance](https://plotly.com/python/reference/layout/#layout-spikedistance)*Code:* fig.update\_layout(spikedistance=<VALUE>)  
  Type: integer greater than or equal to -1  
  Default: -1
* Sets the default distance (in pixels) to look for data to draw spikelines to (-1 means no cutoff, 0 means no looking for data). As with hoverdistance, distance does not apply to area-like objects. In addition, some objects can be hovered on but will not generate spikelines, such as scatter fills.
* [hoverlabel](https://plotly.com/python/reference/layout/#layout-hoverlabel)*Code:* fig.update\_layout(hoverlabel=dict(...))  
  Type: dict containing one or more of the keys listed below.
  + [align](https://plotly.com/python/reference/layout/#layout-hoverlabel-align)*Code:* fig.update\_layout(hoverlabel\_align=<VALUE>)  
    Type: enumerated , one of ( "left" | "right" | "auto" )  
    Default: "auto"
  + Sets the horizontal alignment of the text content within hover label box. Has an effect only if the hover label text spans more two or more lines
  + [bgcolor](https://plotly.com/python/reference/layout/#layout-hoverlabel-bgcolor)*Code:* fig.update\_layout(hoverlabel\_bgcolor=<VALUE>)  
    Type: color
  + Sets the background color of all hover labels on graph
  + [bordercolor](https://plotly.com/python/reference/layout/#layout-hoverlabel-bordercolor)*Code:* fig.update\_layout(hoverlabel\_bordercolor=<VALUE>)  
    Type: color
  + Sets the border color of all hover labels on graph.
  + [font](https://plotly.com/python/reference/layout/#layout-hoverlabel-font)*Code:* fig.update\_layout(hoverlabel\_font=dict(...))  
    Type: dict containing one or more of the keys listed below.
  + Sets the default hover label font used by all traces on the graph.
    - [color](https://plotly.com/python/reference/layout/#layout-hoverlabel-font-color)*Code:* fig.update\_layout(hoverlabel\_font\_color=<VALUE>)  
      Type: color
    - [family](https://plotly.com/python/reference/layout/#layout-hoverlabel-font-family)*Code:* fig.update\_layout(hoverlabel\_font\_family=<VALUE>)  
      Type: string  
      Default: "Arial, sans-serif"
    - HTML font family - the typeface that will be applied by the web browser. The web browser will only be able to apply a font if it is available on the system which it operates. Provide multiple font families, separated by commas, to indicate the preference in which to apply fonts if they aren't available on the system. The Chart Studio Cloud (at <https://chart-studio.plotly.com> or on-premise) generates images on a server, where only a select number of fonts are installed and supported. These include "Arial", "Balto", "Courier New", "Droid Sans",, "Droid Serif", "Droid Sans Mono", "Gravitas One", "Old Standard TT", "Open Sans", "Overpass", "PT Sans Narrow", "Raleway", "Times New Roman".
    - [size](https://plotly.com/python/reference/layout/#layout-hoverlabel-font-size)*Code:* fig.update\_layout(hoverlabel\_font\_size=<VALUE>)  
      Type: number greater than or equal to 1  
      Default: 13
  + [grouptitlefont](https://plotly.com/python/reference/layout/#layout-hoverlabel-grouptitlefont)*Code:* fig.update\_layout(hoverlabel\_grouptitlefont=dict(...))  
    Type: dict containing one or more of the keys listed below.
  + Sets the font for group titles in hover (unified modes). Defaults to `hoverlabel.font`.
    - [color](https://plotly.com/python/reference/layout/#layout-hoverlabel-grouptitlefont-color)*Code:* fig.update\_layout(hoverlabel\_grouptitlefont\_color=<VALUE>)  
      Type: color
    - [family](https://plotly.com/python/reference/layout/#layout-hoverlabel-grouptitlefont-family)*Code:* fig.update\_layout(hoverlabel\_grouptitlefont\_family=<VALUE>)  
      Type: string
    - HTML font family - the typeface that will be applied by the web browser. The web browser will only be able to apply a font if it is available on the system which it operates. Provide multiple font families, separated by commas, to indicate the preference in which to apply fonts if they aren't available on the system. The Chart Studio Cloud (at <https://chart-studio.plotly.com> or on-premise) generates images on a server, where only a select number of fonts are installed and supported. These include "Arial", "Balto", "Courier New", "Droid Sans",, "Droid Serif", "Droid Sans Mono", "Gravitas One", "Old Standard TT", "Open Sans", "Overpass", "PT Sans Narrow", "Raleway", "Times New Roman".
    - [size](https://plotly.com/python/reference/layout/#layout-hoverlabel-grouptitlefont-size)*Code:* fig.update\_layout(hoverlabel\_grouptitlefont\_size=<VALUE>)  
      Type: number greater than or equal to 1
  + [namelength](https://plotly.com/python/reference/layout/#layout-hoverlabel-namelength)*Code:* fig.update\_layout(hoverlabel\_namelength=<VALUE>)  
    Type: integer greater than or equal to -1  
    Default: 15
  + Sets the default length (in number of characters) of the trace name in the hover labels for all traces. -1 shows the whole name regardless of length. 0-3 shows the first 0-3 characters, and an integer >3 will show the whole name if it is less than that many characters, but if it is longer, will truncate to `namelength - 3` characters and add an ellipsis.
* [transition](https://plotly.com/python/reference/layout/#layout-transition)*Code:* fig.update\_layout(transition=dict(...))  
  Type: dict containing one or more of the keys listed below.
* Sets transition options used during Plotly.react updates.
  + [duration](https://plotly.com/python/reference/layout/#layout-transition-duration)*Code:* fig.update\_layout(transition\_duration=<VALUE>)  
    Type: number greater than or equal to 0  
    Default: 500
  + The duration of the transition, in milliseconds. If equal to zero, updates are synchronous.
  + [easing](https://plotly.com/python/reference/layout/#layout-transition-easing)*Code:* fig.update\_layout(transition\_easing=<VALUE>)  
    Type: enumerated , one of ( "linear" | "quad" | "cubic" | "sin" | "exp" | "circle" | "elastic" | "back" | "bounce" | "linear-in" | "quad-in" | "cubic-in" | "sin-in" | "exp-in" | "circle-in" | "elastic-in" | "back-in" | "bounce-in" | "linear-out" | "quad-out" | "cubic-out" | "sin-out" | "exp-out" | "circle-out" | "elastic-out" | "back-out" | "bounce-out" | "linear-in-out" | "quad-in-out" | "cubic-in-out" | "sin-in-out" | "exp-in-out" | "circle-in-out" | "elastic-in-out" | "back-in-out" | "bounce-in-out" )  
    Default: "cubic-in-out"
  + The easing function used for the transition
  + [ordering](https://plotly.com/python/reference/layout/#layout-transition-ordering)*Code:* fig.update\_layout(transition\_ordering=<VALUE>)  
    Type: enumerated , one of ( "layout first" | "traces first" )  
    Default: "layout first"
  + Determines whether the figure's layout or traces smoothly transitions during updates that make both traces and layout change.
* [datarevision](https://plotly.com/python/reference/layout/#layout-datarevision)*Code:* fig.update\_layout(datarevision=<VALUE>)  
  Type: number or categorical coordinate string
* If provided, a changed value tells `Plotly.react` that one or more data arrays has changed. This way you can modify arrays in-place rather than making a complete new copy for an incremental change. If NOT provided, `Plotly.react` assumes that data arrays are being treated as immutable, thus any data array with a different identity from its predecessor contains new data.
* [uirevision](https://plotly.com/python/reference/layout/#layout-uirevision)*Code:* fig.update\_layout(uirevision=<VALUE>)  
  Type: number or categorical coordinate string
* Used to allow user interactions with the plot to persist after `Plotly.react` calls that are unaware of these interactions. If `uirevision` is omitted, or if it is given and it changed from the previous `Plotly.react` call, the exact new figure is used. If `uirevision` is truthy and did NOT change, any attribute that has been affected by user interactions and did not receive a different value in the new figure will keep the interaction value. `layout.uirevision` attribute serves as the default for `uirevision` attributes in various sub-containers. For finer control you can set these sub-attributes directly. For example, if your app separately controls the data on the x and y axes you might set `xaxis.uirevision="time"` and `yaxis.uirevision="cost"`. Then if only the y data is changed, you can update `yaxis.uirevision="quantity"` and the y axis range will reset but the x axis range will retain any user-driven zoom.
* [editrevision](https://plotly.com/python/reference/layout/#layout-editrevision)*Code:* fig.update\_layout(editrevision=<VALUE>)  
  Type: number or categorical coordinate string
* Controls persistence of user-driven changes in `editable: True` configuration, other than trace names and axis titles. Defaults to `layout.uirevision`.
* [selectionrevision](https://plotly.com/python/reference/layout/#layout-selectionrevision)*Code:* fig.update\_layout(selectionrevision=<VALUE>)  
  Type: number or categorical coordinate string
* Controls persistence of user-driven changes in selected points from all traces.
* [template](https://plotly.com/python/reference/layout/#layout-template)*Code:* fig.update\_layout(template=<VALUE>)  
  Type: number or categorical coordinate string
* Default attributes to be applied to the plot. Templates can be created from existing plots using `Plotly.makeTemplate`, or created manually. They should be objects with format: `{layout: layoutTemplate, data: {[type]: [traceTemplate, ...]}, ...}` `layoutTemplate` and `traceTemplate` are objects matching the attribute structure of `layout` and a data trace. Trace templates are applied cyclically to traces of each type. Container arrays (eg `annotations`) have special handling: An object ending in `defaults` (eg `annotationdefaults`) is applied to each array item. But if an item has a `templateitemname` key we look in the template array for an item with matching `name` and apply that instead. If no matching `name` is found we mark the item invisible. Any named template item not referenced is appended to the end of the array, so you can use this for a watermark annotation or a logo image, for example. To omit one of these items on the plot, make an item with matching `templateitemname` and `visible: False`.
* [meta](https://plotly.com/python/reference/layout/#layout-meta)*Code:* fig.update\_layout(meta=<VALUE>)  
  Type: number or categorical coordinate string
* Assigns extra meta information that can be used in various `text` attributes. Attributes such as the graph, axis and colorbar `title.text`, annotation `text` `trace.name` in legend items, `rangeselector`, `updatemenus` and `sliders` `label` text all support `meta`. One can access `meta` fields using template strings: `%{meta[i]}` where `i` is the index of the `meta` item in question. `meta` can also be an object for example `{key: value}` which can be accessed %{meta[key]}.
* [computed](https://plotly.com/python/reference/layout/#layout-computed)*Code:* fig.update\_layout(computed=<VALUE>)  
  Type: number or categorical coordinate string
* Placeholder for exporting automargin-impacting values namely `margin.t`, `margin.b`, `margin.l` and `margin.r` in "full-json" mode.
* [grid](https://plotly.com/python/reference/layout/#layout-grid)*Code:* fig.update\_layout(grid=dict(...))  
  Type: dict containing one or more of the keys listed below.
  + [columns](https://plotly.com/python/reference/layout/#layout-grid-columns)*Code:* fig.update\_layout(grid\_columns=<VALUE>)  
    Type: integer greater than or equal to 1
  + The number of columns in the grid. If you provide a 2D `subplots` array, the length of its longest row is used as the default. If you give an `xaxes` array, its length is used as the default. But it's also possible to have a different length, if you want to leave a row at the end for non-cartesian subplots.
  + [domain](https://plotly.com/python/reference/layout/#layout-grid-domain)*Code:* fig.update\_layout(grid\_domain=dict(...))  
    Type: dict containing one or more of the keys listed below.
    - [x](https://plotly.com/python/reference/layout/#layout-grid-domain-x)*Code:* fig.update\_layout(grid\_domain\_x=list(...))  
      Type: list  
      Default: [0, 1]
    - Sets the horizontal domain of this grid subplot (in plot fraction). The first and last cells end exactly at the domain edges, with no grout around the edges.
    - [y](https://plotly.com/python/reference/layout/#layout-grid-domain-y)*Code:* fig.update\_layout(grid\_domain\_y=list(...))  
      Type: list  
      Default: [0, 1]
    - Sets the vertical domain of this grid subplot (in plot fraction). The first and last cells end exactly at the domain edges, with no grout around the edges.
  + [pattern](https://plotly.com/python/reference/layout/#layout-grid-pattern)*Code:* fig.update\_layout(grid\_pattern=<VALUE>)  
    Type: enumerated , one of ( "independent" | "coupled" )  
    Default: "coupled"
  + If no `subplots`, `xaxes`, or `yaxes` are given but we do have `rows` and `columns`, we can generate defaults using consecutive axis IDs, in two ways: "coupled" gives one x axis per column and one y axis per row. "independent" uses a new xy pair for each cell, left-to-right across each row then iterating rows according to `roworder`.
  + [roworder](https://plotly.com/python/reference/layout/#layout-grid-roworder)*Code:* fig.update\_layout(grid\_roworder=<VALUE>)  
    Type: enumerated , one of ( "top to bottom" | "bottom to top" )  
    Default: "top to bottom"
  + Is the first row the top or the bottom? Note that columns are always enumerated from left to right.
  + [rows](https://plotly.com/python/reference/layout/#layout-grid-rows)*Code:* fig.update\_layout(grid\_rows=<VALUE>)  
    Type: integer greater than or equal to 1
  + The number of rows in the grid. If you provide a 2D `subplots` array or a `yaxes` array, its length is used as the default. But it's also possible to have a different length, if you want to leave a row at the end for non-cartesian subplots.
  + [subplots](https://plotly.com/python/reference/layout/#layout-grid-subplots)*Code:* fig.update\_layout(grid\_subplots=list(...))  
    Type: list
  + Used for freeform grids, where some axes may be shared across subplots but others are not. Each entry should be a cartesian subplot id, like "xy" or "x3y2", or "" to leave that cell empty. You may reuse x axes within the same column, and y axes within the same row. Non-cartesian subplots and traces that support `domain` can place themselves in this grid separately using the `gridcell` attribute.
  + [xaxes](https://plotly.com/python/reference/layout/#layout-grid-xaxes)*Code:* fig.update\_layout(grid\_xaxes=list(...))  
    Type: list
  + Used with `yaxes` when the x and y axes are shared across columns and rows. Each entry should be an x axis id like "x", "x2", etc., or "" to not put an x axis in that column. Entries other than "" must be unique. Ignored if `subplots` is present. If missing but `yaxes` is present, will generate consecutive IDs.
  + [xgap](https://plotly.com/python/reference/layout/#layout-grid-xgap)*Code:* fig.update\_layout(grid\_xgap=<VALUE>)  
    Type: number between or equal to 0 and 1
  + Horizontal space between grid cells, expressed as a fraction of the total width available to one cell. Defaults to 0.1 for coupled-axes grids and 0.2 for independent grids.
  + [xside](https://plotly.com/python/reference/layout/#layout-grid-xside)*Code:* fig.update\_layout(grid\_xside=<VALUE>)  
    Type: enumerated , one of ( "bottom" | "bottom plot" | "top plot" | "top" )  
    Default: "bottom plot"
  + Sets where the x axis labels and titles go. "bottom" means the very bottom of the grid. "bottom plot" is the lowest plot that each x axis is used in. "top" and "top plot" are similar.
  + [yaxes](https://plotly.com/python/reference/layout/#layout-grid-yaxes)*Code:* fig.update\_layout(grid\_yaxes=list(...))  
    Type: list
  + Used with `yaxes` when the x and y axes are shared across columns and rows. Each entry should be an y axis id like "y", "y2", etc., or "" to not put a y axis in that row. Entries other than "" must be unique. Ignored if `subplots` is present. If missing but `xaxes` is present, will generate consecutive IDs.
  + [ygap](https://plotly.com/python/reference/layout/#layout-grid-ygap)*Code:* fig.update\_layout(grid\_ygap=<VALUE>)  
    Type: number between or equal to 0 and 1
  + Vertical space between grid cells, expressed as a fraction of the total height available to one cell. Defaults to 0.1 for coupled-axes grids and 0.3 for independent grids.
  + [yside](https://plotly.com/python/reference/layout/#layout-grid-yside)*Code:* fig.update\_layout(grid\_yside=<VALUE>)  
    Type: enumerated , one of ( "left" | "left plot" | "right plot" | "right" )  
    Default: "left plot"
  + Sets where the y axis labels and titles go. "left" means the very left edge of the grid. "left plot" is the leftmost plot that each y axis is used in. "right" and "right plot" are similar.
* [calendar](https://plotly.com/python/reference/layout/#layout-calendar)*Code:* fig.update\_layout(calendar=<VALUE>)  
  Type: enumerated , one of ( "chinese" | "coptic" | "discworld" | "ethiopian" | "gregorian" | "hebrew" | "islamic" | "jalali" | "julian" | "mayan" | "nanakshahi" | "nepali" | "persian" | "taiwan" | "thai" | "ummalqura" )  
  Default: "gregorian"
* Sets the default calendar system to use for interpreting and displaying dates throughout the plot.
* [minreducedheight](https://plotly.com/python/reference/layout/#layout-minreducedheight)*Code:* fig.update\_layout(minreducedheight=<VALUE>)  
  Type: number greater than or equal to 2  
  Default: 64
* Minimum height of the plot with margin.automargin applied (in px)
* [minreducedwidth](https://plotly.com/python/reference/layout/#layout-minreducedwidth)*Code:* fig.update\_layout(minreducedwidth=<VALUE>)  
  Type: number greater than or equal to 2  
  Default: 64
* Minimum width of the plot with margin.automargin applied (in px)
* [newshape](https://plotly.com/python/reference/layout/#layout-newshape)*Code:* fig.update\_layout(newshape=dict(...))  
  Type: dict containing one or more of the keys listed below.
  + [drawdirection](https://plotly.com/python/reference/layout/#layout-newshape-drawdirection)*Code:* fig.update\_layout(newshape\_drawdirection=<VALUE>)  
    Type: enumerated , one of ( "ortho" | "horizontal" | "vertical" | "diagonal" )  
    Default: "diagonal"
  + When `dragmode` is set to "drawrect", "drawline" or "drawcircle" this limits the drag to be horizontal, vertical or diagonal. Using "diagonal" there is no limit e.g. in drawing lines in any direction. "ortho" limits the draw to be either horizontal or vertical. "horizontal" allows horizontal extend. "vertical" allows vertical extend.
  + [fillcolor](https://plotly.com/python/reference/layout/#layout-newshape-fillcolor)*Code:* fig.update\_layout(newshape\_fillcolor=<VALUE>)  
    Type: color  
    Default: "rgba(0,0,0,0)"
  + Sets the color filling new shapes' interior. Please note that if using a fillcolor with alpha greater than half, drag inside the active shape starts moving the shape underneath, otherwise a new shape could be started over.
  + [fillrule](https://plotly.com/python/reference/layout/#layout-newshape-fillrule)*Code:* fig.update\_layout(newshape\_fillrule=<VALUE>)  
    Type: enumerated , one of ( "evenodd" | "nonzero" )  
    Default: "evenodd"
  + Determines the path's interior. For more info please visit <https://developer.mozilla.org/en-US/docs/Web/SVG/Attribute/fill-rule>
  + [layer](https://plotly.com/python/reference/layout/#layout-newshape-layer)*Code:* fig.update\_layout(newshape\_layer=<VALUE>)  
    Type: enumerated , one of ( "below" | "above" )  
    Default: "above"
  + Specifies whether new shapes are drawn below or above traces.
  + [line](https://plotly.com/python/reference/layout/#layout-newshape-line)*Code:* fig.update\_layout(newshape\_line=dict(...))  
    Type: dict containing one or more of the keys listed below.
    - [color](https://plotly.com/python/reference/layout/#layout-newshape-line-color)*Code:* fig.update\_layout(newshape\_line\_color=<VALUE>)  
      Type: color
    - Sets the line color. By default uses either dark grey or white to increase contrast with background color.
    - [dash](https://plotly.com/python/reference/layout/#layout-newshape-line-dash)*Code:* fig.update\_layout(newshape\_line\_dash=<VALUE>)  
      Type: string  
      Default: "solid"
    - Sets the dash style of lines. Set to a dash type string ("solid", "dot", "dash", "longdash", "dashdot", or "longdashdot") or a dash length list in px (eg "5px,10px,2px,2px").
    - [width](https://plotly.com/python/reference/layout/#layout-newshape-line-width)*Code:* fig.update\_layout(newshape\_line\_width=<VALUE>)  
      Type: number greater than or equal to 0  
      Default: 4
    - Sets the line width (in px).
  + [opacity](https://plotly.com/python/reference/layout/#layout-newshape-opacity)*Code:* fig.update\_layout(newshape\_opacity=<VALUE>)  
    Type: number between or equal to 0 and 1  
    Default: 1
  + Sets the opacity of new shapes.
* [activeshape](https://plotly.com/python/reference/layout/#layout-activeshape)*Code:* fig.update\_layout(activeshape=dict(...))  
  Type: dict containing one or more of the keys listed below.
  + [fillcolor](https://plotly.com/python/reference/layout/#layout-activeshape-fillcolor)*Code:* fig.update\_layout(activeshape\_fillcolor=<VALUE>)  
    Type: color  
    Default: "rgb(255,0,255)"
  + Sets the color filling the active shape' interior.
  + [opacity](https://plotly.com/python/reference/layout/#layout-activeshape-opacity)*Code:* fig.update\_layout(activeshape\_opacity=<VALUE>)  
    Type: number between or equal to 0 and 1  
    Default: 0.5
  + Sets the opacity of the active shape.
* [selections](https://plotly.com/python/reference/layout/#layout-selections)*Code:* fig.update\_layout(selections=list(...))  
  Type: list of dict where each dict has one or more of the keys listed below.
  + [line](https://plotly.com/python/reference/layout/#layout-selections-items-selection-line)*Parent:* layout.selections[]  
    Type: dict containing one or more of the keys listed below.
    - [color](https://plotly.com/python/reference/layout/#layout-selections-items-selection-line-color)*Parent:* layout.selections[].line  
      Type: color
    - Sets the line color.
    - [dash](https://plotly.com/python/reference/layout/#layout-selections-items-selection-line-dash)*Parent:* layout.selections[].line  
      Type: string  
      Default: "dot"
    - Sets the dash style of lines. Set to a dash type string ("solid", "dot", "dash", "longdash", "dashdot", or "longdashdot") or a dash length list in px (eg "5px,10px,2px,2px").
    - [width](https://plotly.com/python/reference/layout/#layout-selections-items-selection-line-width)*Parent:* layout.selections[].line  
      Type: number greater than or equal to 1  
      Default: 1
    - Sets the line width (in px).
  + [name](https://plotly.com/python/reference/layout/#layout-selections-items-selection-name)*Parent:* layout.selections[]  
    Type: string
  + When used in a template, named items are created in the output figure in addition to any items the figure already has in this array. You can modify these items in the output figure by making your own item with `templateitemname` matching this `name` alongside your modifications (including `visible: False` or `enabled: False` to hide it). Has no effect outside of a template.
  + [opacity](https://plotly.com/python/reference/layout/#layout-selections-items-selection-opacity)*Parent:* layout.selections[]  
    Type: number between or equal to 0 and 1  
    Default: 0.7
  + Sets the opacity of the selection.
  + [path](https://plotly.com/python/reference/layout/#layout-selections-items-selection-path)*Parent:* layout.selections[]  
    Type: string
  + For `type` "path" - a valid SVG path similar to `shapes.path` in data coordinates. Allowed segments are: M, L and Z.
  + [templateitemname](https://plotly.com/python/reference/layout/#layout-selections-items-selection-templateitemname)*Parent:* layout.selections[]  
    Type: string
  + Used to refer to a named item in this array in the template. Named items from the template will be created even without a matching item in the input figure, but you can modify one by making an item with `templateitemname` matching its `name`, alongside your modifications (including `visible: False` or `enabled: False` to hide it). If there is no template or no matching item, this item will be hidden unless you explicitly show it with `visible: True`.
  + [type](https://plotly.com/python/reference/layout/#layout-selections-items-selection-type)*Parent:* layout.selections[]  
    Type: enumerated , one of ( "rect" | "path" )
  + Specifies the selection type to be drawn. If "rect", a rectangle is drawn linking (`x0`,`y0`), (`x1`,`y0`), (`x1`,`y1`) and (`x0`,`y1`). If "path", draw a custom SVG path using `path`.
  + [x0](https://plotly.com/python/reference/layout/#layout-selections-items-selection-x0)*Parent:* layout.selections[]  
    Type: number or categorical coordinate string
  + Sets the selection's starting x position.
  + [x1](https://plotly.com/python/reference/layout/#layout-selections-items-selection-x1)*Parent:* layout.selections[]  
    Type: number or categorical coordinate string
  + Sets the selection's end x position.
  + [xref](https://plotly.com/python/reference/layout/#layout-selections-items-selection-xref)*Parent:* layout.selections[]  
    Type: enumerated , one of ( "paper" | "/^x([2-9]|[1-9][0-9]+)?( domain)?$/" )
  + Sets the selection's x coordinate axis. If set to a x axis id (e.g. "x" or "x2"), the `x` position refers to a x coordinate. If set to "paper", the `x` position refers to the distance from the left of the plotting area in normalized coordinates where "0" ("1") corresponds to the left (right). If set to a x axis ID followed by "domain" (separated by a space), the position behaves like for "paper", but refers to the distance in fractions of the domain length from the left of the domain of that axis: e.g., "x2 domain" refers to the domain of the second x axis and a x position of 0.5 refers to the point between the left and the right of the domain of the second x axis.
  + [y0](https://plotly.com/python/reference/layout/#layout-selections-items-selection-y0)*Parent:* layout.selections[]  
    Type: number or categorical coordinate string
  + Sets the selection's starting y position.
  + [y1](https://plotly.com/python/reference/layout/#layout-selections-items-selection-y1)*Parent:* layout.selections[]  
    Type: number or categorical coordinate string
  + Sets the selection's end y position.
  + [yref](https://plotly.com/python/reference/layout/#layout-selections-items-selection-yref)*Parent:* layout.selections[]  
    Type: enumerated , one of ( "paper" | "/^y([2-9]|[1-9][0-9]+)?( domain)?$/" )
  + Sets the selection's x coordinate axis. If set to a y axis id (e.g. "y" or "y2"), the `y` position refers to a y coordinate. If set to "paper", the `y` position refers to the distance from the bottom of the plotting area in normalized coordinates where "0" ("1") corresponds to the bottom (top). If set to a y axis ID followed by "domain" (separated by a space), the position behaves like for "paper", but refers to the distance in fractions of the domain length from the bottom of the domain of that axis: e.g., "y2 domain" refers to the domain of the second y axis and a y position of 0.5 refers to the point between the bottom and the top of the domain of the second y axis.
* [hidesources](https://plotly.com/python/reference/layout/#layout-hidesources)*Code:* fig.update\_layout(hidesources=<VALUE>)  
  Type: boolean
* Determines whether or not a text link citing the data source is placed at the bottom-right cored of the figure. Has only an effect only on graphs that have been generated via forked graphs from the Chart Studio Cloud (at <https://chart-studio.plotly.com> or on-premise).
* [bargap](https://plotly.com/python/reference/layout/#layout-bargap)*Parent:* layout  
  Type: number between or equal to 0 and 1
* Sets the gap (in plot fraction) between bars of adjacent location coordinates.
* [bargroupgap](https://plotly.com/python/reference/layout/#layout-bargroupgap)*Parent:* layout  
  Type: number between or equal to 0 and 1  
  Default: 0
* Sets the gap (in plot fraction) between bars of the same location coordinate.
* [barmode](https://plotly.com/python/reference/layout/#layout-barmode)*Parent:* layout  
  Type: enumerated , one of ( "stack" | "group" | "overlay" | "relative" )  
  Default: "group"
* Determines how bars at the same location coordinate are displayed on the graph. With "stack", the bars are stacked on top of one another With "relative", the bars are stacked on top of one another, with negative values below the axis, positive values above With "group", the bars are plotted next to one another centered around the shared location. With "overlay", the bars are plotted over one another, you might need to reduce "opacity" to see multiple bars.
* [barnorm](https://plotly.com/python/reference/layout/#layout-barnorm)*Parent:* layout  
  Type: enumerated , one of ( "" | "fraction" | "percent" )  
  Default: ""
* Sets the normalization for bar traces on the graph. With "fraction", the value of each bar is divided by the sum of all values at that location coordinate. "percent" is the same but multiplied by 100 to show percentages.
* [extendpiecolors](https://plotly.com/python/reference/layout/#layout-extendpiecolors)*Parent:* layout  
  Type: boolean  
  Default: True
* If `True`, the pie slice colors (whether given by `piecolorway` or inherited from `colorway`) will be extended to three times its original length by first repeating every color 20% lighter then each color 20% darker. This is intended to reduce the likelihood of reusing the same color when you have many slices, but you can set `False` to disable. Colors provided in the trace, using `marker.colors`, are never extended.
* [hiddenlabels](https://plotly.com/python/reference/layout/#layout-hiddenlabels)*Parent:* layout  
  Type: list, numpy array, or Pandas series of numbers, strings, or datetimes.
* hiddenlabels is the funnelarea & pie chart analog of visible:'legendonly' but it can contain many labels, and can simultaneously hide slices from several pies/funnelarea charts
* [piecolorway](https://plotly.com/python/reference/layout/#layout-piecolorway)*Parent:* layout  
  Type: colorlist
* Sets the default pie slice colors. Defaults to the main `colorway` used for trace colors. If you specify a new list here it can still be extended with lighter and darker colors, see `extendpiecolors`.
* [boxgap](https://plotly.com/python/reference/layout/#layout-boxgap)*Parent:* layout  
  Type: number between or equal to 0 and 1  
  Default: 0.3
* Sets the gap (in plot fraction) between boxes of adjacent location coordinates. Has no effect on traces that have "width" set.
* [boxgroupgap](https://plotly.com/python/reference/layout/#layout-boxgroupgap)*Parent:* layout  
  Type: number between or equal to 0 and 1  
  Default: 0.3
* Sets the gap (in plot fraction) between boxes of the same location coordinate. Has no effect on traces that have "width" set.
* [boxmode](https://plotly.com/python/reference/layout/#layout-boxmode)*Parent:* layout  
  Type: enumerated , one of ( "group" | "overlay" )  
  Default: "overlay"
* Determines how boxes at the same location coordinate are displayed on the graph. If "group", the boxes are plotted next to one another centered around the shared location. If "overlay", the boxes are plotted over one another, you might need to set "opacity" to see them multiple boxes. Has no effect on traces that have "width" set.
* [violingap](https://plotly.com/python/reference/layout/#layout-violingap)*Parent:* layout  
  Type: number between or equal to 0 and 1  
  Default: 0.3
* Sets the gap (in plot fraction) between violins of adjacent location coordinates. Has no effect on traces that have "width" set.
* [violingroupgap](https://plotly.com/python/reference/layout/#layout-violingroupgap)*Parent:* layout  
  Type: number between or equal to 0 and 1  
  Default: 0.3
* Sets the gap (in plot fraction) between violins of the same location coordinate. Has no effect on traces that have "width" set.
* [violinmode](https://plotly.com/python/reference/layout/#layout-violinmode)*Parent:* layout  
  Type: enumerated , one of ( "group" | "overlay" )  
  Default: "overlay"
* Determines how violins at the same location coordinate are displayed on the graph. If "group", the violins are plotted next to one another centered around the shared location. If "overlay", the violins are plotted over one another, you might need to set "opacity" to see them multiple violins. Has no effect on traces that have "width" set.
* [bargap](https://plotly.com/python/reference/layout/#layout-bargap)*Parent:* layout  
  Type: number between or equal to 0 and 1
* Sets the gap (in plot fraction) between bars of adjacent location coordinates.
* [bargroupgap](https://plotly.com/python/reference/layout/#layout-bargroupgap)*Parent:* layout  
  Type: number between or equal to 0 and 1  
  Default: 0
* Sets the gap (in plot fraction) between bars of the same location coordinate.
* [barmode](https://plotly.com/python/reference/layout/#layout-barmode)*Parent:* layout  
  Type: enumerated , one of ( "stack" | "group" | "overlay" | "relative" )  
  Default: "group"
* Determines how bars at the same location coordinate are displayed on the graph. With "stack", the bars are stacked on top of one another With "relative", the bars are stacked on top of one another, with negative values below the axis, positive values above With "group", the bars are plotted next to one another centered around the shared location. With "overlay", the bars are plotted over one another, you might need to reduce "opacity" to see multiple bars.
* [barnorm](https://plotly.com/python/reference/layout/#layout-barnorm)*Parent:* layout  
  Type: enumerated , one of ( "" | "fraction" | "percent" )  
  Default: ""
* Sets the normalization for bar traces on the graph. With "fraction", the value of each bar is divided by the sum of all values at that location coordinate. "percent" is the same but multiplied by 100 to show percentages.
* [boxgap](https://plotly.com/python/reference/layout/#layout-boxgap)*Parent:* layout  
  Type: number between or equal to 0 and 1  
  Default: 0.3
* Sets the gap (in plot fraction) between boxes of adjacent location coordinates. Has no effect on traces that have "width" set.
* [boxgroupgap](https://plotly.com/python/reference/layout/#layout-boxgroupgap)*Parent:* layout  
  Type: number between or equal to 0 and 1  
  Default: 0.3
* Sets the gap (in plot fraction) between boxes of the same location coordinate. Has no effect on traces that have "width" set.
* [boxmode](https://plotly.com/python/reference/layout/#layout-boxmode)*Parent:* layout  
  Type: enumerated , one of ( "group" | "overlay" )  
  Default: "overlay"
* Determines how boxes at the same location coordinate are displayed on the graph. If "group", the boxes are plotted next to one another centered around the shared location. If "overlay", the boxes are plotted over one another, you might need to set "opacity" to see them multiple boxes. Has no effect on traces that have "width" set.
* [waterfallgap](https://plotly.com/python/reference/layout/#layout-waterfallgap)*Parent:* layout  
  Type: number between or equal to 0 and 1
* Sets the gap (in plot fraction) between bars of adjacent location coordinates.
* [waterfallgroupgap](https://plotly.com/python/reference/layout/#layout-waterfallgroupgap)*Parent:* layout  
  Type: number between or equal to 0 and 1  
  Default: 0
* Sets the gap (in plot fraction) between bars of the same location coordinate.
* [waterfallmode](https://plotly.com/python/reference/layout/#layout-waterfallmode)*Parent:* layout  
  Type: enumerated , one of ( "group" | "overlay" )  
  Default: "group"
* Determines how bars at the same location coordinate are displayed on the graph. With "group", the bars are plotted next to one another centered around the shared location. With "overlay", the bars are plotted over one another, you might need to reduce "opacity" to see multiple bars.
* [funnelgap](https://plotly.com/python/reference/layout/#layout-funnelgap)*Parent:* layout  
  Type: number between or equal to 0 and 1
* Sets the gap (in plot fraction) between bars of adjacent location coordinates.
* [funnelgroupgap](https://plotly.com/python/reference/layout/#layout-funnelgroupgap)*Parent:* layout  
  Type: number between or equal to 0 and 1  
  Default: 0
* Sets the gap (in plot fraction) between bars of the same location coordinate.
* [funnelmode](https://plotly.com/python/reference/layout/#layout-funnelmode)*Parent:* layout  
  Type: enumerated , one of ( "stack" | "group" | "overlay" )  
  Default: "stack"
* Determines how bars at the same location coordinate are displayed on the graph. With "stack", the bars are stacked on top of one another With "group", the bars are plotted next to one another centered around the shared location. With "overlay", the bars are plotted over one another, you might need to reduce "opacity" to see multiple bars.
* [extendfunnelareacolors](https://plotly.com/python/reference/layout/#layout-extendfunnelareacolors)*Parent:* layout  
  Type: boolean  
  Default: True
* If `True`, the funnelarea slice colors (whether given by `funnelareacolorway` or inherited from `colorway`) will be extended to three times its original length by first repeating every color 20% lighter then each color 20% darker. This is intended to reduce the likelihood of reusing the same color when you have many slices, but you can set `False` to disable. Colors provided in the trace, using `marker.colors`, are never extended.
* [funnelareacolorway](https://plotly.com/python/reference/layout/#layout-funnelareacolorway)*Parent:* layout  
  Type: colorlist
* Sets the default funnelarea slice colors. Defaults to the main `colorway` used for trace colors. If you specify a new list here it can still be extended with lighter and darker colors, see `extendfunnelareacolors`.
* [hiddenlabels](https://plotly.com/python/reference/layout/#layout-hiddenlabels)*Parent:* layout  
  Type: list, numpy array, or Pandas series of numbers, strings, or datetimes.
* hiddenlabels is the funnelarea & pie chart analog of visible:'legendonly' but it can contain many labels, and can simultaneously hide slices from several pies/funnelarea charts
* [bargap](https://plotly.com/python/reference/layout/#layout-bargap)*Parent:* layout  
  Type: number between or equal to 0 and 1  
  Default: 0.1
* Sets the gap between bars of adjacent location coordinates. Values are unitless, they represent fractions of the minimum difference in bar positions in the data.
* [barmode](https://plotly.com/python/reference/layout/#layout-barmode)*Parent:* layout  
  Type: enumerated , one of ( "stack" | "overlay" )  
  Default: "stack"
* Determines how bars at the same location coordinate are displayed on the graph. With "stack", the bars are stacked on top of one another With "overlay", the bars are plotted over one another, you might need to reduce "opacity" to see multiple bars.
* [extendsunburstcolors](https://plotly.com/python/reference/layout/#layout-extendsunburstcolors)*Parent:* layout  
  Type: boolean  
  Default: True
* If `True`, the sunburst slice colors (whether given by `sunburstcolorway` or inherited from `colorway`) will be extended to three times its original length by first repeating every color 20% lighter then each color 20% darker. This is intended to reduce the likelihood of reusing the same color when you have many slices, but you can set `False` to disable. Colors provided in the trace, using `marker.colors`, are never extended.
* [sunburstcolorway](https://plotly.com/python/reference/layout/#layout-sunburstcolorway)*Parent:* layout  
  Type: colorlist
* Sets the default sunburst slice colors. Defaults to the main `colorway` used for trace colors. If you specify a new list here it can still be extended with lighter and darker colors, see `extendsunburstcolors`.
* [extendtreemapcolors](https://plotly.com/python/reference/layout/#layout-extendtreemapcolors)*Parent:* layout  
  Type: boolean  
  Default: True
* If `True`, the treemap slice colors (whether given by `treemapcolorway` or inherited from `colorway`) will be extended to three times its original length by first repeating every color 20% lighter then each color 20% darker. This is intended to reduce the likelihood of reusing the same color when you have many slices, but you can set `False` to disable. Colors provided in the trace, using `marker.colors`, are never extended.
* [treemapcolorway](https://plotly.com/python/reference/layout/#layout-treemapcolorway)*Parent:* layout  
  Type: colorlist
* Sets the default treemap slice colors. Defaults to the main `colorway` used for trace colors. If you specify a new list here it can still be extended with lighter and darker colors, see `extendtreemapcolors`.
* [extendiciclecolors](https://plotly.com/python/reference/layout/#layout-extendiciclecolors)*Parent:* layout  
  Type: boolean  
  Default: True
* If `True`, the icicle slice colors (whether given by `iciclecolorway` or inherited from `colorway`) will be extended to three times its original length by first repeating every color 20% lighter then each color 20% darker. This is intended to reduce the likelihood of reusing the same color when you have many slices, but you can set `False` to disable. Colors provided in the trace, using `marker.colors`, are never extended.
* [iciclecolorway](https://plotly.com/python/reference/layout/#layout-iciclecolorway)*Parent:* layout  
  Type: colorlist
* Sets the default icicle slice colors. Defaults to the main `colorway` used for trace colors. If you specify a new list here it can still be extended with lighter and darker colors, see `extendiciclecolors`.