Amtiri

REFERENCE MANUAL

Gauges

Generated on November 25, 2026

Four widgets show a reading against a range: a dial, the scale that goes around it, and two straight bars. They are SVG shapes on the canvas, drawn in the order of the widget list like every other shape.

Widget	Palette tile	Stored type	Draws	Typical use
Radial gauge	Radial Gauge	`RadialGauge`	An arc from one value to another on a dial, over a track	Pressures, speeds, a level shown as a dial
Radial labels	Radial Labels	`RadialLabels`	Texts spread along an arc	The scale of a radial gauge
Horizontal gauge	Horizontal Gauge	`HorizontalGauge`	A bar from one value to another, left to right	Flows, loads, the progress of a batch
Vertical gauge	Vertical Gauge	`VerticalGauge`	A bar from one value to another, bottom to top	Levels, temperatures

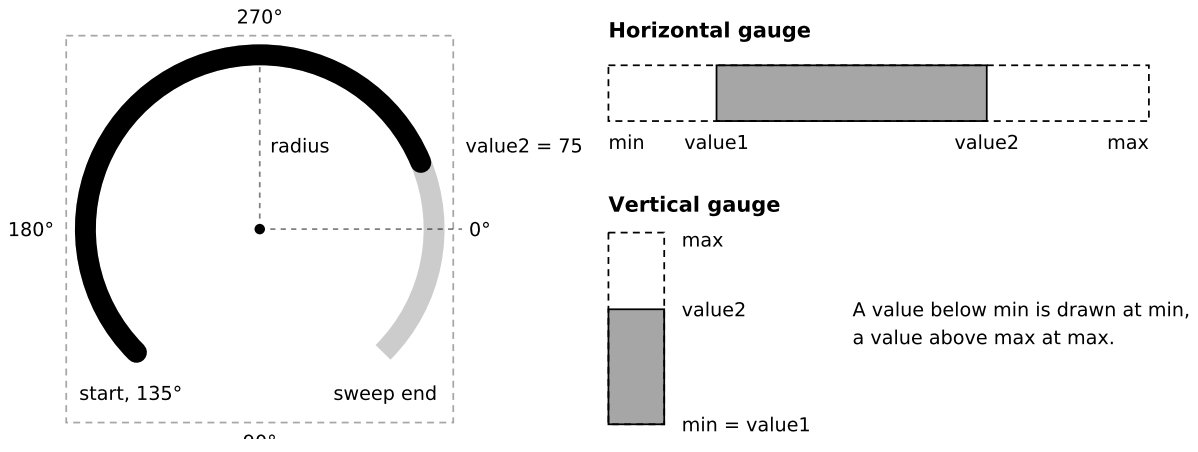
The gauges also have the properties every widget shares: see Properties every widget shares.

How a gauge reads its values

A gauge has fixed values, such as `minValue` and `value2`, and a formula for each, such as Min value and Value 2 in its setup panel. The two work differently from the offsets of other widgets:

- **A formula replaces its fixed value.** Value 2 set to `LIT_301` draws the level itself; nothing is added to the fixed value. An offset such as `dX` on a rectangle is added to its position instead.
- **An empty or failing formula falls back to the fixed value.** The fixed values have no field in the setup panel, so a new gauge keeps the ones it was created with: a range from 0 to 100, Value 1 at 0 and Value 2 at 75.
- **The editor evaluates the formulas too**, so a gauge shows the plant's live reading while the dashboard is being edited. The offsets `dX`, `dY`, `dWidth` and `dHeight` are still evaluated only in the viewer.

Every gauge turns a value into a share of its range the same way, and draws between the shares of Value 1 and Value 2:



90°

share = (value − min) / (max − min), kept between 0 and 1.
Radial: angle = startAngle + sweepAngle × share, with 0° pointing right and angles growing clockwise.
Linear: the bar covers the part of the box between the shares of value1 and value2.

- **Value 1 and Value 2** mark the two ends of what is drawn. For a reading, leave Value 1 at the minimum and set Value 2 to the reading. For a band, such as an operating window, set both.
- **Readings off the range** are drawn at its ends: a level above the maximum shows a full gauge, never one that overflows its box.

Radial gauge

A radial gauge draws a dial centred in its box. The **track** is the whole arc, from Start angle over Sweep angle; the **value arc** runs over it, from the angle of Value 1 to the angle of Value 2. A new gauge is 80 × 80 px, with a track from 135° over 270°: it opens at the bottom, as most dials do. The track is drawn with `styles`, and the value arc with Value styles.

- **Angles** are in degrees. 0° points right, towards three o'clock, and angles grow clockwise: 90° points down and 270° up. A negative Sweep angle runs the dial anticlockwise.
- **Radius** is a share of half the box's shorter side: 0.9, the default, puts the arc just inside the box. The arcs' strokes are centred on the radius, so half their width lies outside it. At the default 80 × 80 px with an 8 px value arc, the outer edge of the stroke touches the box: 0.9 × 40 + 4 = 40 px.
- **Keep Min value and Max value apart.** With both equal there is no range to place a value in, and the value arc shows either nothing or the whole track.
- **Offsets.** The setup panel has no fields for `dX`, `dY`, `dWidth` and `dHeight`; place and size the gauge with its handles.
- A gauge compiles its Value styles again on every repaint of the view. On a screen with many gauges and fast-changing values, keep Value styles short.

Properties

Key	Editor label (English)	Formula or value	Result type	Default	Meaning

`minValue`	None: no editor	Value	Number	`0`	The value at the start of the track, used when Min value is empty or fails
`maxValue`	None: no editor	Value	Number	`100`	The value at the end of the track, used when Max value is empty or fails
`value1`	None: no editor	Value	Number	`0`	Where the value arc starts, used when Value 1 is empty or fails
`value2`	None: no editor	Value	Number	`75`	Where the value arc ends, used when Value 2 is empty or fails
`startAngle`	None: no editor	Value	Number, degrees	`135`	Where the track starts, used when Start angle is empty or fails
`sweepAngle`	None: no editor	Value	Number, degrees	`270`	How far the track runs, used when Sweep angle is empty or fails
`radius`	None: no editor	Value	Number, share of half the shorter side	`0.9`	The arcs' radius, used when Radius is empty or fails
`dMinValue`	Min value	Formula; replaces `minValue`; the editor evaluates it too	Number	Empty	The value at the start of the track
`dMaxValue`	Max value	Formula; replaces `maxValue`; the editor evaluates it too	Number	Empty	The value at the end of the track
`dValue1`	Value 1	Formula; replaces `value1`; the editor evaluates it too	Number	Empty	Where the value arc starts
`dValue2`	Value 2	Formula; replaces `value2`; the editor evaluates it too	Number	Empty	Where the value arc ends
`dStartAngle`	Start angle	Formula; replaces `startAngle`; the editor evaluates it too	Number, degrees	Empty	Where the track starts; 0° points right
`dSweepAngle`	Sweep angle	Formula; replaces `sweepAngle`; the editor evaluates it too	Number, degrees	Empty	How far the track runs, clockwise when positive
`dRadius`	Radius	Formula; replaces `radius`; the editor evaluates it too	Number, share of half the shorter side	Empty	The arcs' radius
`valueStyles`	Value styles	Formula; the editor evaluates it too	Record	`{fill:"none",stroke:"#0CF",stroke_width:8,stroke_linecap:"round"}`	The styles of the value arc

Example

The level of a 4.5 m tank, red near the top:

```
Min value      0
Max value      4.5
Value 1        0
Value 2        LIT_301
Value styles   {fill:"none",stroke:if(LIT_301 > 4.2,"#DC2626","#0EA5E9"),stroke_width:10,stroke_linecap:"round"
```

At 3.6 m the value arc runs from 135° to 135° + 270° × 0.8 = 351°.

Styling

- **Elements**: an SVG `<g>` holding two `<polyline>`s: the track, styled by `styles`, and the value arc over it, styled by `valueStyles`.
- **Default styles**: a new gauge's `styles` is `{fill:"none",stroke:"#DDD",stroke_width:8,stroke_linecap:"round"}`, a light grey track as wide as the value arc, with round ends. A gauge whose Styles field is empty, as on every gauge saved before the track was drawn, draws no track: such a dial is usually drawn by other widgets under the gauge. `valueStyles` is `{fill:"none",stroke:"#0CF",stroke_width:8,stroke_linecap:"round"}`.
- **Keys that work**, on both: `stroke`, `stroke_width`, `stroke_opacity`, `stroke_linecap` (`round`, `butt` or `square`), `stroke_dasharray` (as text, `"4 2"`) and `opacity`.
- **Keys to avoid**: any `fill` but `"none"`: a fill paints the area between the arc and the straight line joining its ends.

The widget reads no style key itself.

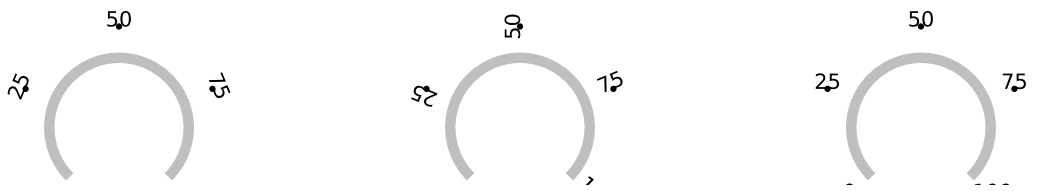
Styled example. A flat dial: a wide grey track with square ends, and a value arc of the same width over it:

```
Styles         {fill:"none",stroke:"#E5E7EB",stroke_width:12,stroke_linecap:"butt"}
Value styles   {fill:"none",stroke:"#2563EB",stroke_width:12,stroke_linecap:"butt"}
```

Radial labels

Radial labels draw the texts of their `labels` list along an arc centred in their box: label i of n sits at the angle Start angle + Sweep angle × i/(n # 1), so the first is at the start of the arc and the last at its end. A list of one item puts it at the middle of the arc. The distance from the centre is Radius, as a share of half the box's shorter side, plus Text offset in pixels. A new widget is 80 × 80 px, with the angles and radius of a new radial gauge.

- **A scale for a gauge.** Give the labels the gauge's box, Start angle and Sweep angle, and n labels for the values min + (max # min) × i/(n # 1): five labels `[0, 25, 50, 75, 100]` on a gauge from 0 to 100 fall on those values. A positive Text offset moves them outside the gauge's arc, a negative one inside.
- **Each label stands on its point**, as a Label does: its baseline goes through the point, and it is centred on it by default.
- **Rotation.** With Rotate labels ticked, the default, each label is turned around its point by its angle plus Rotation offset:





Rotation offset 90 (default)
along the arc, tops outward



Rotation offset 0
along the radius, reading outward



Rotate labels off
upright

- With the rotation offset at 90 the labels on the lower half of a dial read upside down; untick Rotate labels for a scale that reads upright all round.
- **The list** holds texts or numbers, shown as their text as on a Label. A formula that fails draws no label.
- **Offsets.** The setup panel has no fields for `dX`, `dY`, `dWidth` and `dHeight`; place and size the widget with its handles.

Properties

Key	Editor label (English)	Formula or value	Result type	Default	Meaning
`labels`	Labels	Formula; the editor evaluates it too	List of texts or numbers	`["L1", "L2", "L3", "L4"]`	The texts, from the start of the arc to its end
`startAngle`	None: no editor	Value	Number, degrees	`135`	Where the arc starts, used when Start angle is empty or fails
`sweepAngle`	None: no editor	Value	Number, degrees	`270`	How far the arc runs, used when Sweep angle is empty or fails
`radius`	None: no editor	Value	Number, share of half the shorter side	`0.9`	The labels' distance from the centre, used when Radius is empty or fails
`textOffset`	None: no editor	Value	Number, px	`0`	Added to the radius, used when Text offset is empty or fails
`rotateLabels`	Rotate labels	Value, a checkbox	Boolean	`true`	Turns each label by its angle plus the rotation offset; unticked, the labels stay upright
`rotationOffset`	None: no editor	Value	Number, degrees	`90`	Added to each label's angle to turn it, used when Rotation offset is empty or fails
`dStartAngle`	Start angle	Formula; replaces `startAngle`; the editor evaluates it too	Number, degrees	Empty	Where the arc starts; 0° points right
`dSweepAngle`	Sweep angle	Formula; replaces `sweepAngle`; the editor evaluates it too	Number, degrees	Empty	How far the arc runs, clockwise when positive
`dRadius`	Radius	Formula; replaces `radius`; the editor evaluates it too	Number, share of half the shorter side	Empty	The labels' distance from the centre
`dTextOffset`	Text offset	Formula; replaces `textOffset`; the editor evaluates it too	Number, px	Empty	Pixels added to the radius: positive outward, negative inward
`dRotationOffset`	Rotation offset	Formula; replaces `rotationOffset`; the editor evaluates it too	Number, degrees	Empty	Added to each label's angle to turn it: 90 lays it along the arc, 0 along the radius

Example

The scale of the tank gauge above, upright and inside its arc:

```
Labels          [0, 1.5, 3, 4.5]
Text offset     -16
Rotate labels   unticked
Styles          {fill:"#374151",font_size:"11px",text_anchor:"middle"}
```

The four labels fall on 0, 1.5, 3 and 4.5 m of a gauge with the same box and angles and a range from 0 to 4.5.

Styling

- **Element**: an SVG `<g>` holding one `<text>` per label, each turned by a `transform` attribute when Rotate labels is ticked. `styles` applies to every `<text>` alike, so one label cannot be coloured apart from the others.
- **Default styles**: `{fill:"#000",font_size:"14px",text_anchor:"middle"}`.
- **Keys that work**: as on a Label: `fill`, `stroke`, `stroke_width`, `font_family`, `font_size`, `font_weight`, `font_style`, `letter_spacing`, `text_anchor` and `opacity`.
- **Keys to avoid**: `transform`, which replaces the rotation of every label.

The widget reads two style keys itself:

Key	What the widget does with it
`font_size`	Sizes the texts. Without it, the texts are 12 px
`text_anchor`	`start`, `middle` or `end`: the part of each text that sits on its point

Styled example. Bold white figures on a dark dial:

```
Styles   {fill:"#FFFFFF",font_size:"12px",font_weight:"bold",text_anchor:"middle"}
```

Horizontal gauge

A horizontal gauge draws one bar across the full height of its box, covering the part of the box between Value 1 and Value 2, with the minimum at the left edge. A new one is 160 × 32 px.

- **The bar is drawn inside the box** after the offsets have moved and resized it: `dX`, `dY`, `dWidth` and `dHeight` work as on any box, in the viewer.
- **Value 1 above Value 2** still draws the bar between them.
- **Keep Min value and Max value apart**: with both equal the gauge draws no bar.
- **There is no track.** Draw one with a Rectangle of the same box, placed after the gauge in the widget list so that the bar is drawn over it.
- **`styles` is not applied.** The setup panel offers no Styles field: the bar takes its look from Value styles only.

Properties

Key	Editor label (English)	Formula or value	Result type	Default	Meaning
`minValue`	None: no editor	Value	Number	`0`	The value at the left edge, used when Min value is empty or fails
`maxValue`	None: no editor	Value	Number	`100`	The value at the right edge, used when Max value is empty or fails
`value1`	None: no editor	Value	Number	`0`	One end of the bar, used when Value 1 is empty or fails
`value2`	None: no editor	Value	Number	`75`	The other end of the bar, used when Value 2 is empty or fails
`dMinValue`	Min value	Formula; replaces `minValue`; the editor evaluates it too	Number	Empty	The value at the left edge
`dMaxValue`	Max value	Formula; replaces `maxValue`; the editor evaluates it too	Number	Empty	The value at the right edge
`dValue1`	Value 1	Formula; replaces `value1`; the editor evaluates it too	Number	Empty	One end of the bar
`dValue2`	Value 2	Formula; replaces `value2`; the editor evaluates it too	Number	Empty	The other end of the bar
`valueStyles`	Value styles	Formula; the editor evaluates it too	Record	`{fill:"#0CF",stroke:"#0CF",stroke_width:2,rx:"16px",ry:"16px"}`	The styles of the bar

Example

A pump's flow against its range of 0 to 120 m³/h, amber above 100:

```
Max value      120
Value 2        FIT_501
Value styles   {fill:if(FIT_501 > 100,"#F59E0B","#10B981"),stroke:"none",rx:"6px",ry:"6px"}
```

The track under it is a rectangle with the same box, placed after the gauge:

```
Styles   {fill:"#E5E7EB",stroke:"none",rx:"6px",ry:"6px"}
```

Styling

- **Element**: one SVG `<rect>`, the bar. `valueStyles` applies to it; `styles` is not applied.
- **Default styles**: `valueStyles` is `{fill:"#0CF",stroke:"#0CF",stroke_width:2,rx:"16px",ry:"16px"}`. At the new gauge's 32 px thickness that is a capsule; on a thicker box the corners stay 16 px.
- **Keys that work**: `fill`, `fill_opacity`, `stroke`, `stroke_width`, `stroke_opacity`, `rx`, `ry` and `opacity`. The stroke is drawn centred on the bar's edge, half of it outside; `stroke_width:0` draws a bar exactly between the two values.
- **Keys to avoid**: `x`, `y`, `width` and `height`, which some browsers take as the bar's geometry and draw instead of it.
- `rx` and `ry` are lengths: a number is written in pixels, and a text must carry its unit, as `"6px"` does.

The widget reads no style key itself.

Styled example. A square-ended bar with a thin dark outline:

```
Value styles   {fill:"#93C5FD",stroke:"#1E3A8A",stroke_width:1,rx:0,ry:0}
```

Vertical gauge

A vertical gauge draws one bar across the full width of its box, covering the part of the box between Value 1 and Value 2, with the minimum at the bottom edge. A new one is 32 × 160 px. Everything else works as on a Horizontal gauge: the offsets, the order of the values, the missing track and the ignored `styles`.

Properties

Key	Editor label (English)	Formula or value	Result type	Default	Meaning
`minValue`	None: no editor	Value	Number	`0`	The value at the bottom edge, used when Min value is empty or fails
`maxValue`	None: no editor	Value	Number	`100`	The value at the top edge, used when Max value is empty or fails
`value1`	None: no editor	Value	Number	`0`	One end of the bar, used when Value 1 is empty or fails
`value2`	None: no editor	Value	Number	`75`	The other end of the bar, used when Value 2 is empty or fails
`dMinValue`	Min value	Formula; replaces `minValue`; the editor evaluates it too	Number	Empty	The value at the bottom edge
`dMaxValue`	Max value	Formula; replaces `maxValue`; the editor evaluates it too	Number	Empty	The value at the top edge
`dValue1`	Value 1	Formula; replaces `value1`; the editor evaluates it too	Number	Empty	One end of the bar
`dValue2`	Value 2	Formula; replaces `value2`; the editor evaluates it too	Number	Empty	The other end of the bar
`valueStyles`	Value styles	Formula; the editor evaluates it too	Record	`{fill:"#0CF",stroke:"#0CF",stroke_width:2,rx:"16px",ry:"16px"}`	The styles of the bar

Example

A tank's level from 0 to 4.5 m, with a band from the low alarm to the high alarm drawn by a second gauge:

```
Level gauge
  Max value      4.5
  Value 2        LIT_301
  Value styles   {fill:"#38BDF8",stroke:"none",rx:0,ry:0}

Band gauge
  Max value      4.5
  Value 1        LIT_301_LL
  Value 2        LIT_301_HH
  Value styles   {fill:"#22C55E",fill_opacity:0.25,stroke:"none",rx:0,ry:0}
```

Give both gauges the same box, and place the band gauge after the level gauge, so that the level is drawn over the band.

Styling

- **Element**: one SVG `<rect>`, the bar. `valueStyles` applies to it; `styles` is not applied.
- **Default styles**: `valueStyles` is `{fill:"#0CF",stroke:"#0CF",stroke_width:2,rx:"16px",ry:"16px"}`. At the new gauge's 32 px width that is a capsule; on a wider box the corners stay 16 px.
- **Keys that work**: `fill`, `fill_opacity`, `stroke`, `stroke_width`, `stroke_opacity`, `rx`, `ry` and `opacity`. The stroke is drawn centred on the bar's edge, half of it outside; `stroke_width:0` draws a bar exactly between the two values.
- **Keys to avoid**: `x`, `y`, `width` and `height`, as on the horizontal gauge.
- `rx` and `ry` are lengths: a number is written in pixels, and a text must carry its unit.

The widget reads no style key itself.

Styled example. A level bar that turns red above its high alarm:

```
Value styles   {fill:if(LIT_301 > LIT_301_HH,"#DC2626","#38BDF8"),stroke:"none",rx:"4px",ry:"4px"}
```

Next steps

- Charts
- Shapes and text
- Properties every widget shares
- Styling widgets

This page describes Data Orchester Dashboards 1.9.5.