KLay Layered Layout Options

KLay Layered supports a whole bunch of layout options. Every single one of them is documented here.

Contents

Overview

For a general introduction on layout options, see the KIML documentation. KLay Layered supports layout options defined by KIML and defines additional custom layout options.

Supported KIML Layout Options

KLay Layered supports the following standard layout options defined by KIML. Note that the default value may be altered (highlighted yellow). These layout options are documented on KIML's Layout Options page.

Option	ID	Type	Applies to	Default
Additional Port Space	de.cau.cs.kieler.additionalPortSpace	Margins	Nodes	0, 0, 0, 0
Alignment	de.cau.cs.kieler.alignment	Enum	Nodes	AUTOMATIC
Aspect Ratio	de.cau.cs.kieler.aspectRatio	Float	Parents	1.6
Border Spacing	de.cau.cs.kieler.borderSpacing	Float	Parents	20
Comment Box	de.cau.cs.kieler.commentBox	Boolean	Nodes	false
Debug Mode	de.cau.cs.kieler.debugMode	Boolean	Parents	false
Diagram Type	de.cau.cs.kieler.diagramType	String		
Direction	de.cau.cs.kieler.direction	Enum	Parents	UNDEFINED
Edge Label Placement	de.cau.cs.kieler.edgeLabelPlacement	Enum	Labels	UNDEFINED
Edge Routing	de.cau.cs.kieler.edgeRouting	Enum	Parents	ORTHOGONAL
Hypernode	de.cau.cs.kieler.hypernode	Boolean	Nodes	false
Label Side	de.cau.cs.kieler.labelSide	Enum	Parents	SMART
Label Spacing	de.cau.cs.kieler.labelSpacing	Float	Edges Nodes	-1.0
Layout Hierarchy	de.cau.cs.kieler.layoutHierarchy	Boolean	Parents	false
Minimal Height	de.cau.cs.kieler.minHeight	Float	Nodes Parents	0.0
Minimal Width	de.cau.cs.kieler.minWidth	Float	Nodes Parents	0.0
No Layout	de.cau.cs.kieler.noLayout	Boolean		false
Node Label Placement	de.cau.cs.kieler.nodeLabelPlacement	EnumSet	Nodes	

Port Anchor Offset	de.cau.cs.kieler.klay.layered.portAnchor	Object	Ports	
Port Constraints	de.cau.cs.kieler.portConstraints	Enum	Nodes	UNDEFINED
Port Label Placement	de.cau.cs.kieler.portLabelPlacement	Enum	Nodes	OUTSIDE
Port Offset	de.cau.cs.kieler.offset	Float	Ports	
Port Side	de.cau.cs.kieler.portSide	Enum	Ports	UNDEFINED
Port Spacing	de.cau.cs.kieler.portSpacing	Float	Nodes	10
Priority	de.cau.cs.kieler.priority	Int	Edges Nodes	
Randomization Seed	de.cau.cs.kieler.randomSeed	Int	Parents	1
Separate Connected Components	de.cau.cs.kieler.separateConnComp	Boolean	Parents	true
Size Constraint	de.cau.cs.kieler.sizeConstraint	EnumSet	Nodes	
Size Options	de.cau.cs.kieler.sizeOptions	EnumSet	Nodes	DEFAULT_MINIMUM_SIZE
Spacing	de.cau.cs.kieler.spacing	Float	Parents	20

Custom Layout Options

Option	ID	Type	Applies to	Default	Dependency
Add Unnecessary Bendpoints	de.cau.cs.kieler.klay.layered.unnecessaryBendpoints	Boolean	Parents	false	
Content Alignment	de.cau.cs.kieler.klay.layered.contentAlignment	EnumSet	Parents	V_TOP, H_LEFT	
Crossing Minimization	de.cau.cs.kieler.klay.layered.crossMin	Enum	Parents	LAYER_SWEEP	
Cycle Breaking	de.cau.cs.kieler.klay.layered.cycleBreaking	Enum	Parents	GREEDY	
Edge Spacing Factor	de.cau.cs.kieler.klay.layered.edgeSpacingFactor	Float	Parents	0.5	
Edge Label Side Selection	de.cau.cs.kieler.klay.layered.edgeLabelSideSelection	Enum	Parents	ALWAYS_DOWN	
Feedback Edges	de.cau.cs.kieler.klay.layered.feedBackEdges	Boolean	Parents	false	
Fixed Alignment	de.cau.cs.kieler.klay.layered.fixedAlignment	Enum	Parents	NONE	nodePlace=BRANDES_KOEPF
Interactive Reference Point	de.cau.cs.kieler.klay.layered.interactiveReferencePoint	Enum	Parents	CENTER	
Layer Constraint	de.cau.cs.kieler.klay.layered.layerConstraint	Enum	Nodes	NONE	
Linear Segments Deflection Dampening	de.cau.cs.kieler.klay.layered. linearSegmentsDeflectionDampening	Float	Parents	0.3	nodePlace=LINEAR_SEGMENTS
Merge Edges	de.cau.cs.kieler.klay.layered.mergeEdges	Boolean	Parents	false	
Merge Hierarchy-Crossing Edges	de.cau.cs.kieler.klay.layered.mergeHierarchyPorts	Boolean	Parents	true	layoutHierarchy=true
Node Layering	de.cau.cs.kieler.klay.layered.nodeLayering	Enum	Parents	NETWORK_SIMPLEX	
Node Placement	de.cau.cs.kieler.klay.layered.nodePlace	Enum	Parents	BRANDES_KOEPF	
Thoroughness	de.cau.cs.kieler.klay.layered.thoroughness	Int	Parents	10	

Detailed Documentation

This section explains every layout option in more detail. See the KIML documentation for more information on KIML layout options. Those options are only mentioned here if KLay Layered adds some custom behavior.

Add Unnecessary Bendpoints

By default, KLay Layered tries not to add bendpoints to an edge at positions where the edge doesn't change direction since there's no real bend there. Turning this option on forces such bend points. More specifically, a bend point is added for each edge that spans more than one layer at the point where it crosses a layer. If hierarchy layout is turned on, a bend point is also added whenever the edge crosses a hierarchy boundary.

Content Alignment

Determines how the content of compound nodes is to be aligned if the compound node's size exceeds the bounding box of the content (i.e. child nodes). This might be the case if for a compound node the size constraint of `MINIMUM_SIZE` is set and the minimum width and height are set large enough.

⚠ This option is not tested for external ports with port constraints `FIXED_RATIO` or `FIXED_POS`.

Crossing Minimization

Crossing minimization determines the ordering of nodes in each layer, which influences the number of edge crossings. This option switches between one of several algorithms that can be used to minimize crossings. Possible values are:

- `LAYER_SWEEP`
  The layer sweep algorithm iterates multiple times over the layers, trying to find node orderings that minimize the number of crossings. The algorithm uses randomization to increase the odds of finding a good result. To improve its results, consider increasing the *Thoroughness* option, which influences the number of iterations done. The *Randomization* seed also influences results.
- `INTERACTIVE`
  Orders the nodes of each layer by comparing their positions before the layout algorithm was started. The idea is that the relative order of nodes as it was before layout was applied is not changed. This of course requires valid positions for all nodes to have been set on the input graph before calling the layout algorithm. The interactive layer sweep algorithm uses the *Interactive Reference Point* option to determine which reference point of nodes are used to compare positions.

Cycle Breaking

KLay Layered tries to position nodes in a way that all edges point rightwards. This is not possible if the input graph has cycles. Such cycles have to be broken by reversing as few edges as possible. The reversed edges end up pointing leftwards in the resulting diagram. There are different cycle breaking algorithms available:

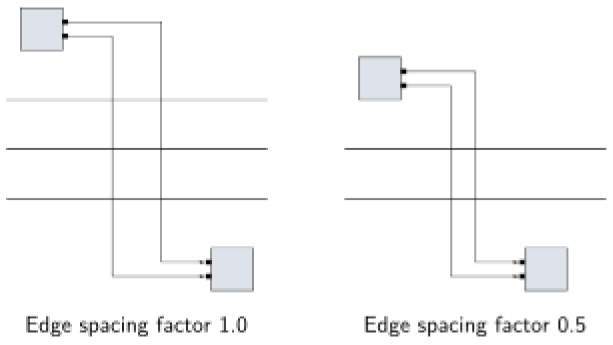
- `GREEDY`
  This algorithm reverses edges greedily. The algorithm tries to avoid edges that have the *Priority* property set.
- `INTERACTIVE`
  The interactive algorithm tries to reverse edges that already pointed leftwards in the input graph. This requires node and port coordinates to have been set to sensible values.

Direction

The layout direction influences where the majority of edges in the final layout will point to. With data flow diagrams, this will usually be to the right. With control flow diagrams, it might be downwards. The layout direction defaults to `UNDEFINED`. This causes KLay Layered to calculate a layout direction based on the `ASPECT_RATIO` setting. As of now, if the aspect ratio is >=1 (that is, if the diagram should be wider than it is high), the direction is set to `RIGHT`. Otherwise, it is set to `DOWN`.

Edge Spacing Factor

The edge spacing factor determines the amount of space between edges, relative to the regular *Spacing* value. The idea is that we don't need as much space between edges as we do between nodes.



Edge Label Side Selection

Determines how KLay Layered places edge labels. The following strategies are available:

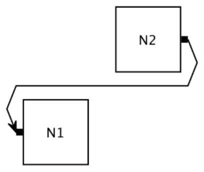
- `ALWAYS_UP`
  Always places edge labels above the edge.
- `ALWAYS_DOWN`
  Always places edge labels below the edge.
- `DIRECTION_UP`
  Places edge labels above edges pointing right, and below edges pointing left.
- `DIRECTION_DOWN`
  Places edge labels below edges pointing right, and above edges pointing left.

- `SMART`
  Uses a heuristic that determines the best edge label placement, also taking the placement of port labels into account.

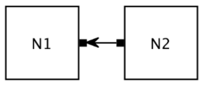
Feedback Edges

Feedback edges are edges that feed the output of a node back to be the input of a previous node. This option controls how feedback edges are routed if port constraints are FREE. This influences how much emphasis is put on feedback edges.

With feedback edges:



Without feedback edges:



Fixed Alignment

The `BRANDES_KOEPF` node placement algorithm computes several different node placements. One of the placements is chosen by the algorithm, usually the one that takes the least amount of space. With this option, a particular result can be chosen.

This option should usually be left alone.

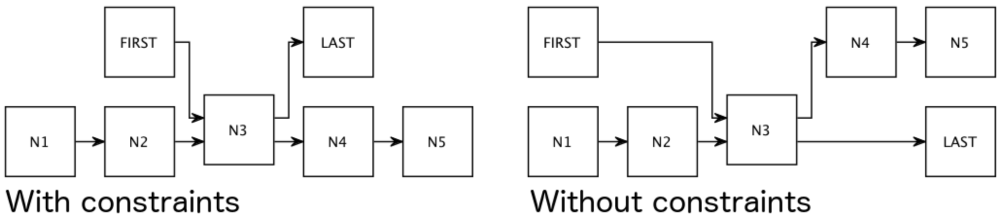
Interactive Reference Point

Interactive layering, crossing minimization, and cycle breaking algorithms use node positions to sort nodes into layers or to determine the order of nodes in each layer. However, it is unclear if for example the top left corners of nodes should be compared, or the bottom left corners — different settings might lead to different results. The interactive reference point determines which part of nodes is used to compare their positions. It provides the following settings:

- `TOP_LEFT`
  The top left corner of a node is taken as the reference point.
- `CENTER`
  The center of a node is taken as the reference point.

Layer Constraint

The layer a node is placed in is usually computed by the layer assignment algorithms. However, sometimes certain nodes need to be placed in the first or in the last layer (for example, nodes that represent inputs from the outside). The layer constraint option can be set on such nodes to do just that.



> ⚠ This option can also be set to `FIRST_SEPARATE` and `LAST_SEPARATE`. These are for internal use only and should not have been publicly exposed in the first place. Using them can result in layout problems.

Linear Segments Deflection Dampening

> ⚠ This is a very advanced layout option that you normally shouldn't worry about.

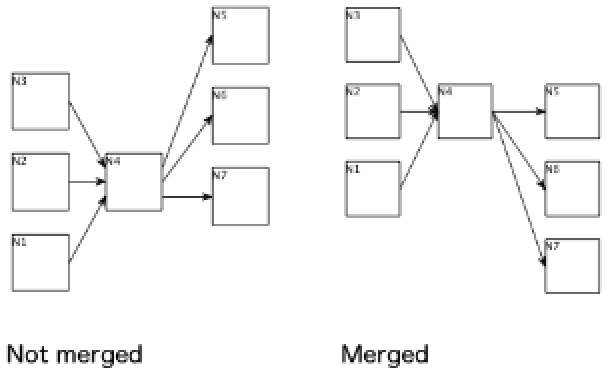
The linear segments node placer can sometimes place nodes in a way that results in unnecessarily large diagrams. This option dampens how much the nodes are moved around. A dampening factor of 1.0 disables dampening and just lets the node placer do what it wants. A more conservative dampening factor of 0.3 (the default) restricts the freedom of the node placer a bit more.

Maximal Iterations

Delimits the amount of depth-first-search iterations performed by the network simplex layering strategy. Large, highly connected graphs might require a long time to be processed. This property serves as a timeout after which an exception is raised.

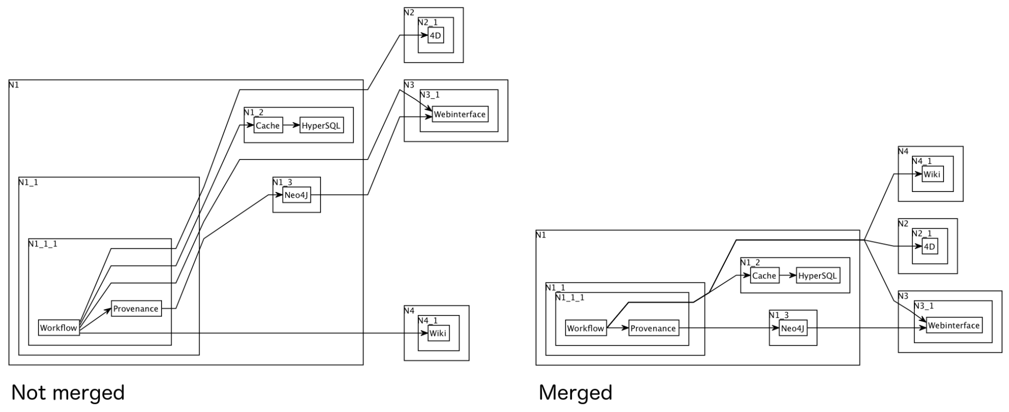
Merge Edges

In the KGraph model, edges can either connect to nodes through ports or directly. In the latter case, KLay Layered will introduce a virtual port for each edge, which results in all edges connecting to the node at different points in the final drawing. If this option is switched on, KLay Layered will only generate up to one input and one output port for each node. The option is set on a parent node and applies to all of its children, but not to the parent node itself.



Merge Hierarchy-Crossing Edges

If hierarchical layout is active, this option is the hierarchical equivalent to *Merge Edges*. If set to true on a compound node, all hierarchy-crossing edges that start or end inside that compound node are eligible for merging.



Node Layering

Decides which algorithm is used to compute the layer each node is placed in. We have different algorithms available, with different optimization goals:

- `NETWORK_SIMPLEX`
  This algorithm tries to minimize the length of edges. This is the most computationally intensive algorithm. The number of iterations after which it aborts if it hasn't found a result yet can be set with the Maximal Iterations option.
- `LONGEST_PATH`
  A very simple algorithm that distributes nodes along their longest path to a sink node.
- `INTERACTIVE`
  Distributes the nodes into layers by comparing their positions before the layout algorithm was started. The idea is that the relative horizontal order of nodes as it was before layout was applied is not changed. This of course requires valid positions for all nodes to have been set on the input graph before calling the layout algorithm. The interactive node layering algorithm uses the *Interactive Reference Point* option to determine which reference point of nodes are used to compare positions.

Node Placement

Decides which algorithm is used to compute the y coordinate of each node. This influences the length of edges, the number of edge bends, and the height of the diagram. We have different algorithms available, with different optimization goals:

- `BRANDES_KOEPF`
  Minimizes the number of edge bends at the expense of diagram size: diagrams drawn with this algorithm are usually higher than diagrams drawn with other algorithms.
- `LINEAR_SEGMENTS`
  Computes a balanced placement.
- `INTERACTIVE`
  Tries to keep the preset y coordinates of nodes from the original layout. For dummy nodes, a guess is made to infer their coordinates. Requires the other interactive phase implementations to have run as well.
- `SIMPLE`
  Minimizes the area at the expense of... well, pretty much everything else.

Thoroughness

There are heuristics in use all over KLay Layered whose results often improve with the number of iterations computed. The thoroughness is a measure for telling KLay Layered to compute more iterations to improve the quality of results, at the expense of performance.