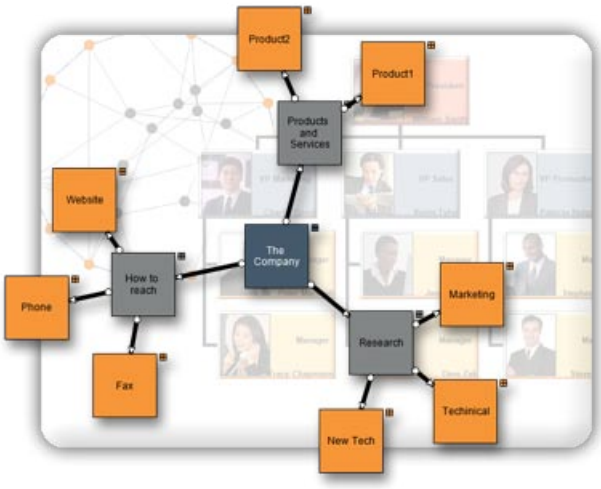
Nevron Diagram for .NET - Layered Graph Layout



Introduction

The layered graph layout algorithm aims to highlight the main direction or flow within a directed graph. Cyclic dependencies of nodes are automatically detected and resolved and nodes are placed in hierarchically arranged layers. Additionally, the ordering of the nodes within each layer is chosen in such a way that the number of edge crossings is small. This layout algorithm is very appropriate for hierarchically organized diagrams, organization charts, etc. Nevron is pleased to present you the *NLayeredGraphLayout* class which implements the layered graph drawing algorithm within Nevron Diagram.

Nevron Diagram algorithm takes as input a simple graph and produces a hierarchically organized drawing of it. The algorithm maintains the following aesthetic criteria (given by their priority in descending order):

- minimal number of levels
- minimal number of edge crossings (a level by level sweep optimization is used)
- minimal number of bends

Implementation

The input of the Layered Graph Layout algorithm must be a directed acyclic graph (DAG), so if the graph contains cycles a preliminary step that removes them is performed. When we ensure that the input graph is a DAG we may perform the first phase of the algorithm - the Layer Distribution. It distributes the vertices of the graph to layers by using the Longest Path Layering algorithm. All vertices without predecessors are put at level 1 and the other vertices are distributed to the next level / levels in such a way that the following statements are true:

- The final graph must occupy as small area as possible
- The layers must be regular. This can be achieved easily by adding dummy vertices for each edge (u, v) with length more than 1 layer
- The number of added dummy vertices must be minimal

When the vertices are distributed to layers we perform a crossing reduction step which rearranges the vertices on each layer in such way that the number of edge crossings is minimized. Finally we calculate the coordinates of the vertices and edges of the drawing and render them using the Nevron Diagram Rendering Engine.

Users Manual

The *NLayeredGraphLayout* class is the class that implements the Nevron Layered Graph Layout algorithm. It layouts the vertices in the graph on hierarchically organized levels. The algorithm is very appropriate to use on all types of graphs (and especially on hierarchically organized ones), except for dense graphs (i.e. graphs with a large number of edges compared to their number of vertices). The result of the layout and the aesthetic criteria for the final drawing can be controlled through the following properties:

- **Direction** – determines how the drawing is oriented in the 2D drawing space. By default it is oriented from Top to Bottom. Other possible orientations are: from Bottom to Top, from Left to Right and from Right to Left.
- **LayerSpacing and VertexSpacing** – determine the spacing between the layers and the vertices.
- **EdgeRouting** – determines whether to use polylines or only horizontal and vertical line segments to draw the edges.

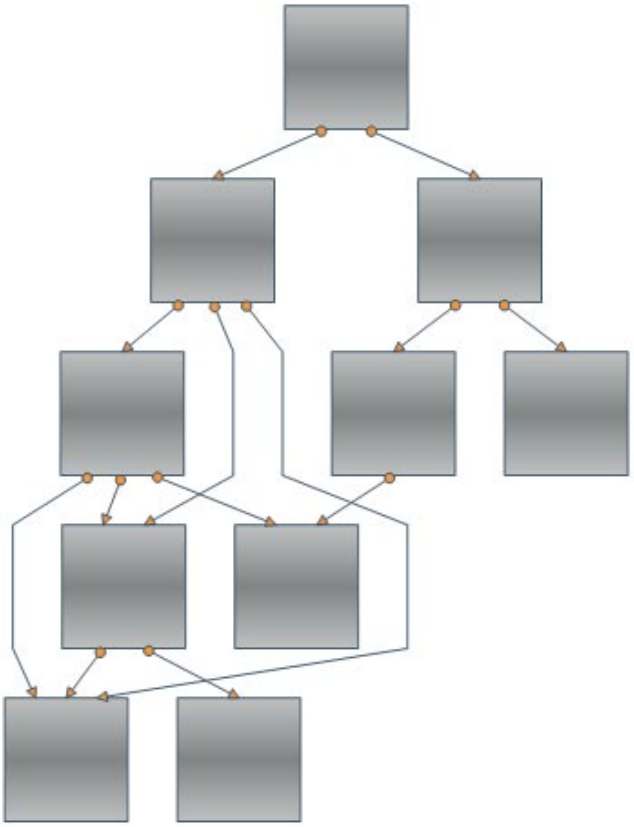


Figure 1. OrthogonalRouting: false

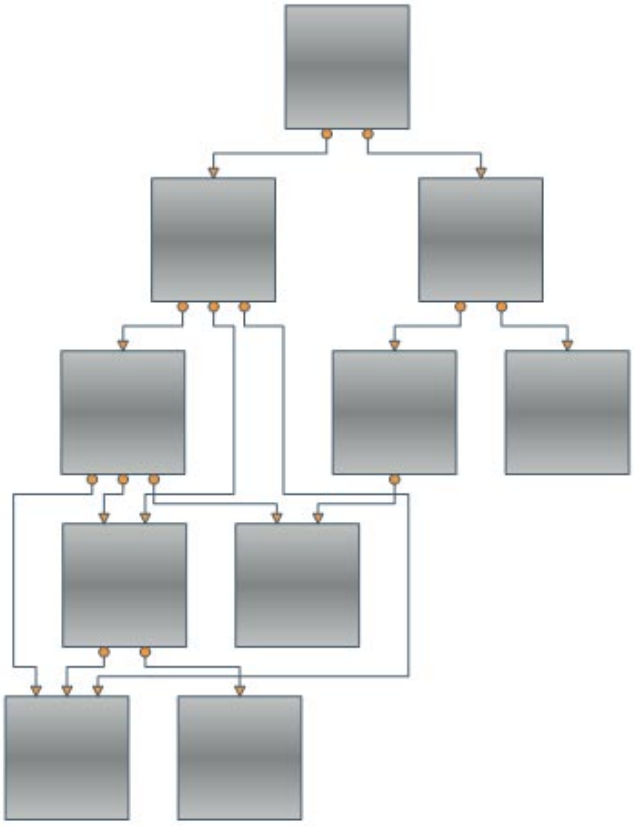


Figure 2. OrthogonalRouting: true

- **InPlugSpacing and OutPlugSpacing** – determine the spacing between the plugs of a node. You can set a fixed amount of spacing or a proportional spacing, which means that the plugs are distributed along the whole side of the node.

- **LayerAlignment and NodeAlignment** – specifies how the vertices are aligned within the layer vertically and horizontally respectively. The possible values are Near, Center and Far. For example if the direction is TopToBottom, then Near means that the vertices are aligned to the top of the layer, Center – to the center of the layer and Far – to the Bottom of the layer.

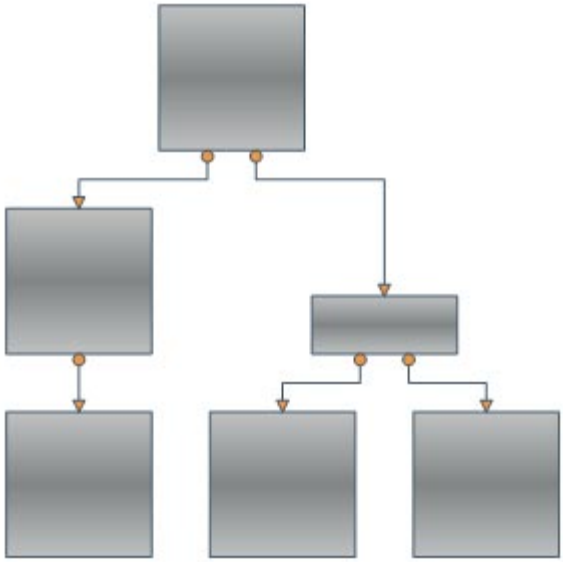


Figure 3. LayerAlignment: Near

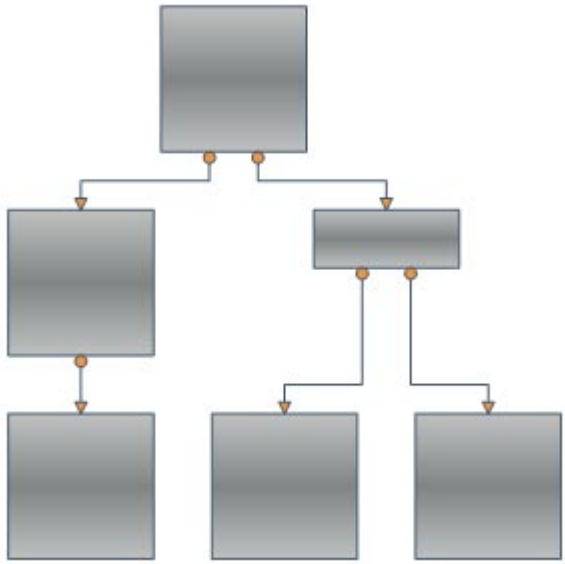


Figure 4. LayerAlignment: Far

- **NodeRank** - specifies the node ranking policy used by the layout. It can be:
  - *TopMost* - all nodes without incoming edges are assigned to the topmost layer
  - *Gravity* - layer distribution is done in such a way that the total length of all edges is minimized
  - *Optimal* - similar to the topmost, but after the initial assignment all nodes fall downwards as much as possible

Conclusion

Nevron Layered Graph Layout is a powerful layout algorithm that produces very nice drawings of hierarchically organized diagrams. The layout is integrated in **Nevron Diagram for .NET** thus giving you the power to layout any diagram in a hierarchical manner using only a few lines of code.

Besides the robust and highly optimized implementation of the Layered Graph Layout, **Nevron Diagram for .NET** provides a dozen of other well implemented and carefully tested layout algorithms. If we take in mind all these layouts, the high number of examples and the detailed programmers' documentation provided by Nevron, we may conclude that the professional visualization of complex graphs and diagrams has never been easier.

Learn more

For more information on Nevron Diagram for .NET and view the demo examples, download a fully functional evaluation of the product, visit: http://www.nevron.com.