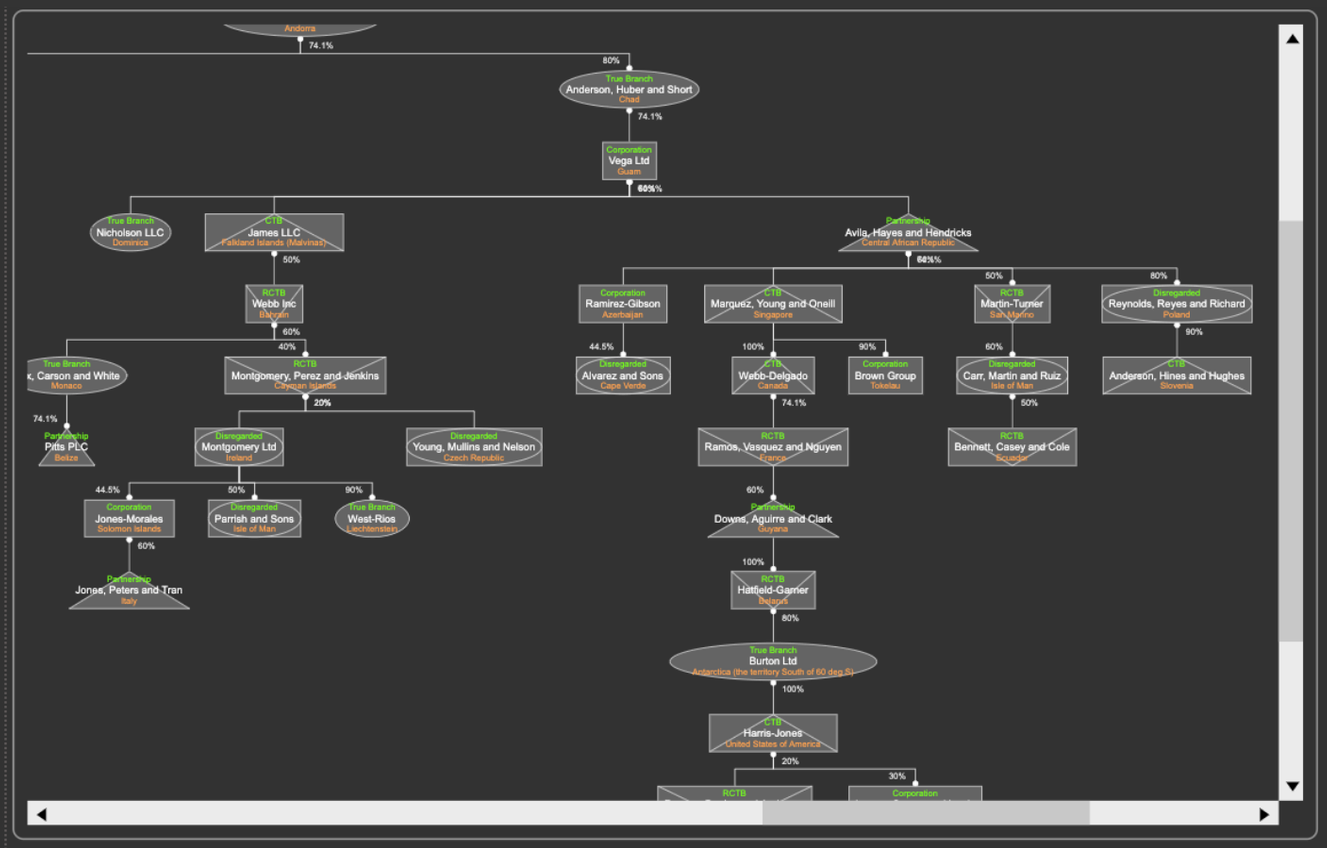
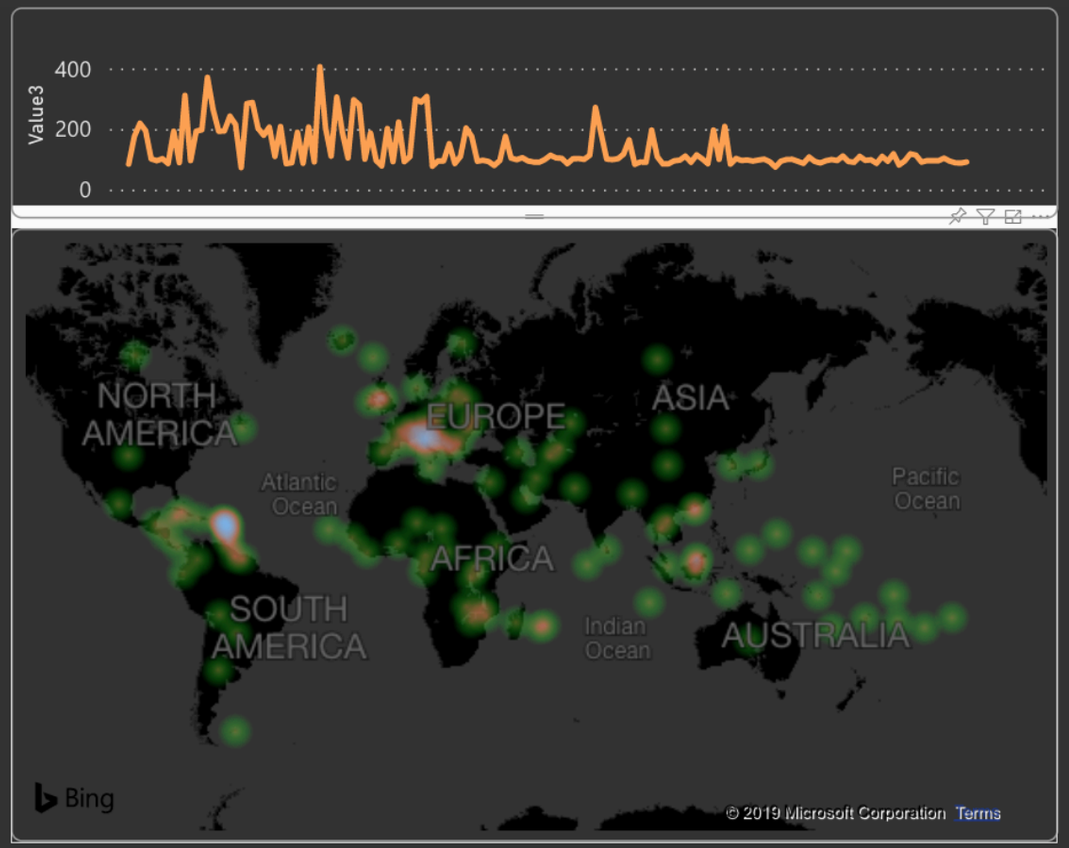
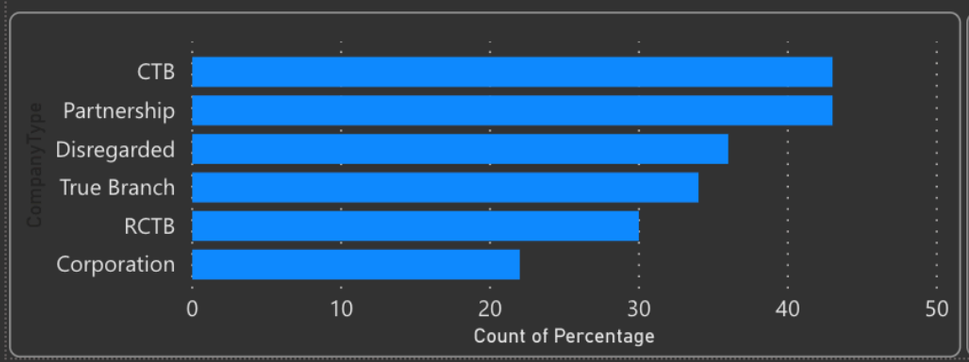
yWorks Consulting

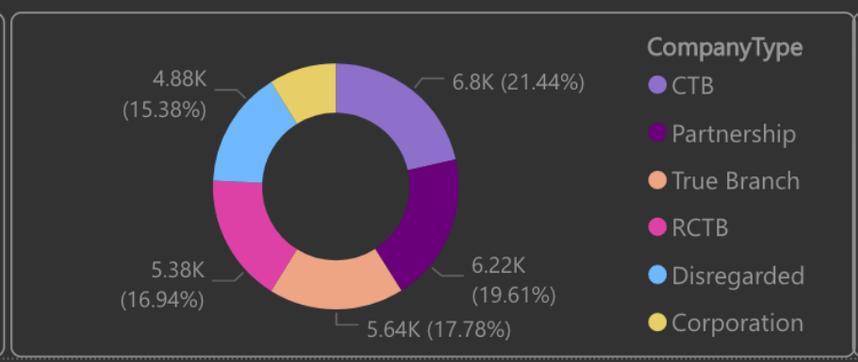
Power BI Widget

A concise overview of the yFiles custom visual for Power BI. An overview of the features, functionality, some technical details and how to use it to create smashing interactive dashboards.

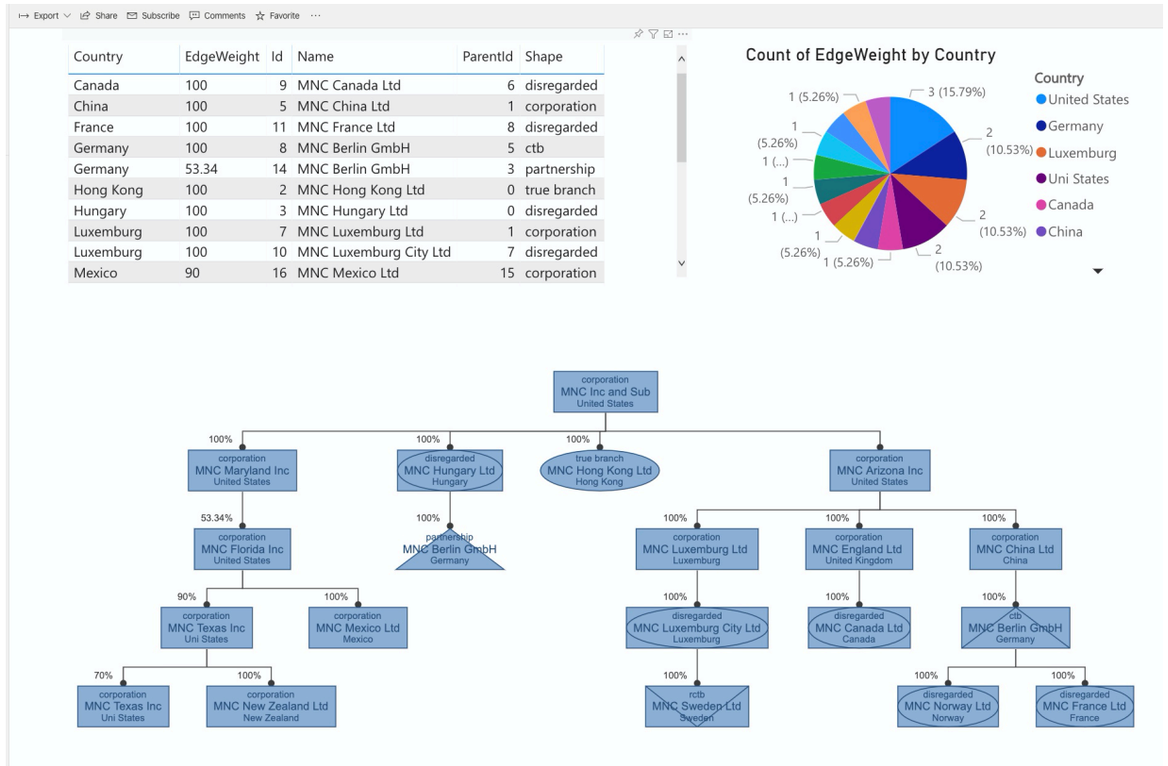






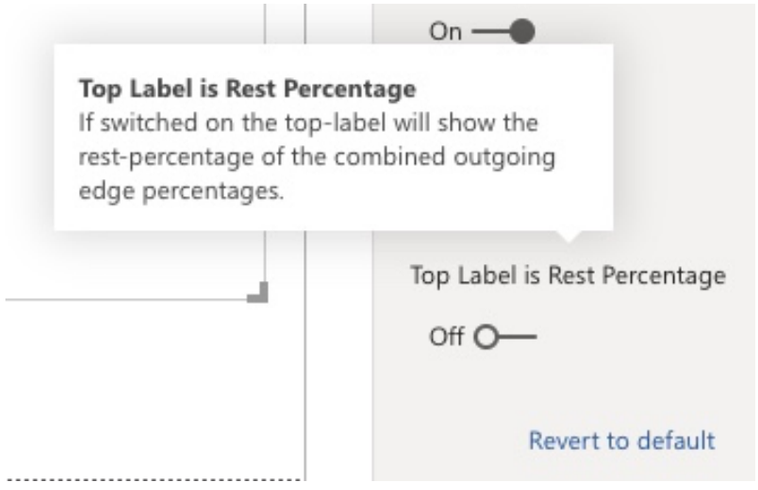


CompanyName	Country
Hernandez and Sons	Japan
Hernandez, Daniels and Hudson	Andorra
Hernandez-Owens	New Caledoni
Hill, White and Douglas	Slovakia (Slova
Howell-Herman	Ukraine



Country	EdgeWeight	Id	Name	ParentId	Shape
Canada	100	9	MNC Canada Ltd	6	disregarded
China	100	5	MNC China Ltd	1	corporation
France	100	11	MNC France Ltd	8	disregarded
Germany	100	8	MNC Berlin GmbH	5	ctb
Germany	53.34	14	MNC Berlin GmbH	3	partnership
Hong Kong	100	2	MNC Hong Kong Ltd	0	true branch
Hungary	100	3	MNC Hungary Ltd	0	disregarded
Luxemburg	100	7	MNC Luxemburg Ltd	1	corporation
Luxemburg	100	10	MNC Luxemburg City Ltd	7	disregarded
Mexico	90	16	MNC Mexico Ltd	15	corporation

There is a special setting in the 'Nodes' section of the property panel to show in the top-label the percentage deficit (delta) of the outgoing edges. This will override any field set for the top-label. Note that if the outgoing values exceed 100% the shown deficit will be negative.



Data Mapping

The yFiles widget consumes a shared dataset. The desktop version of Power BI allows multiple sets but the online only allows one dataset per report. It is via this shared dataset that the widgets within one report can communicate (slicing etc.).

In order to have a graph one needs at least two fields:

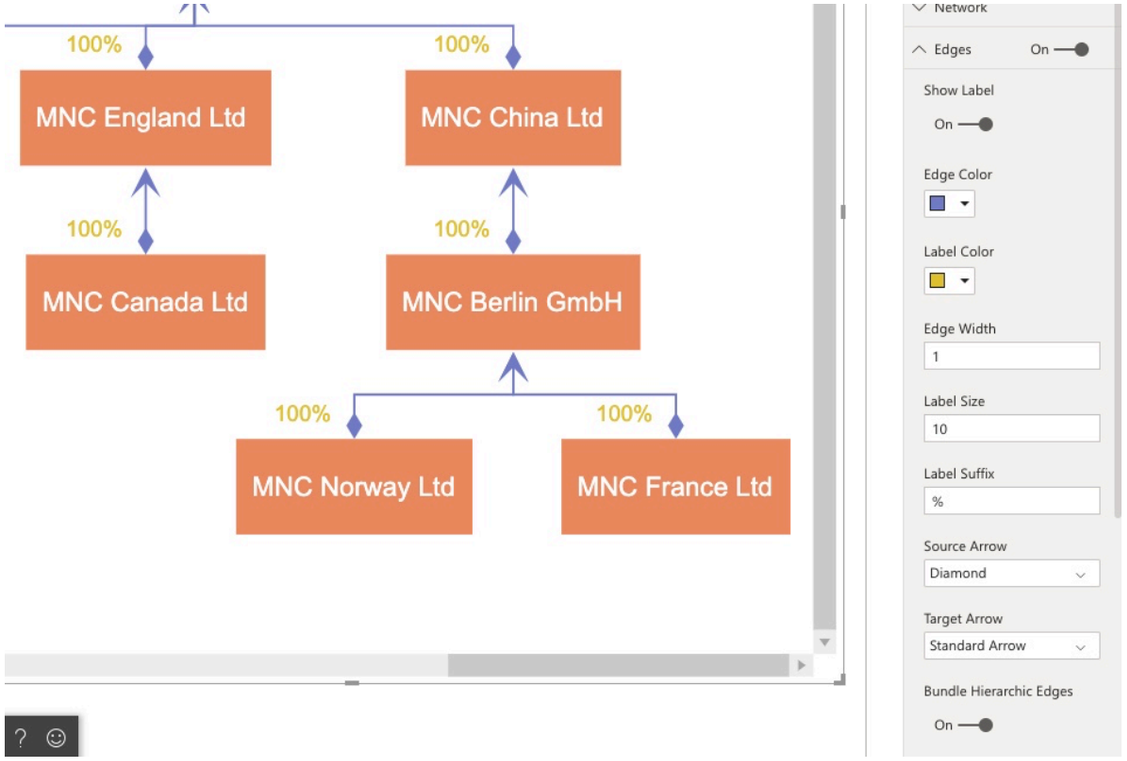
- **NodeId**: an id defining the start of an edge. This can be a unique identifier from an entity (say from a database). Internally this id is converted to a string.
- **TargetId**: an id of another entity defining the sink of an edge.

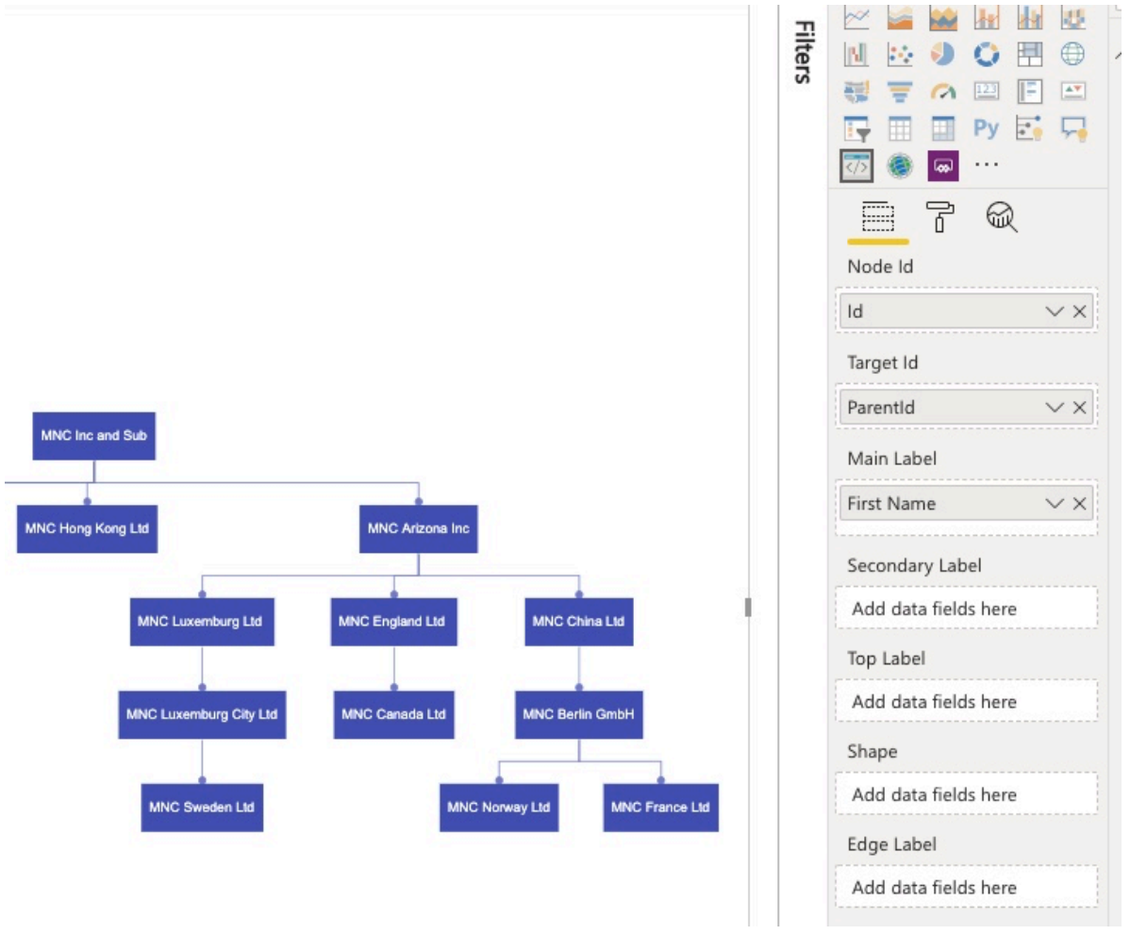
All other fields are optional but you will normally also wish to define:

- the **main label**: the central label shown in the node
- the **shape**: corresponding to the business names given (partnership, ctb, etc.)
- the **edge label**: shown next to the edge, by default at the beginning or source of an edge

In addition, one can also define:

- a **top-label** shown above the main label
- a **sub-label** show underneath the main label.





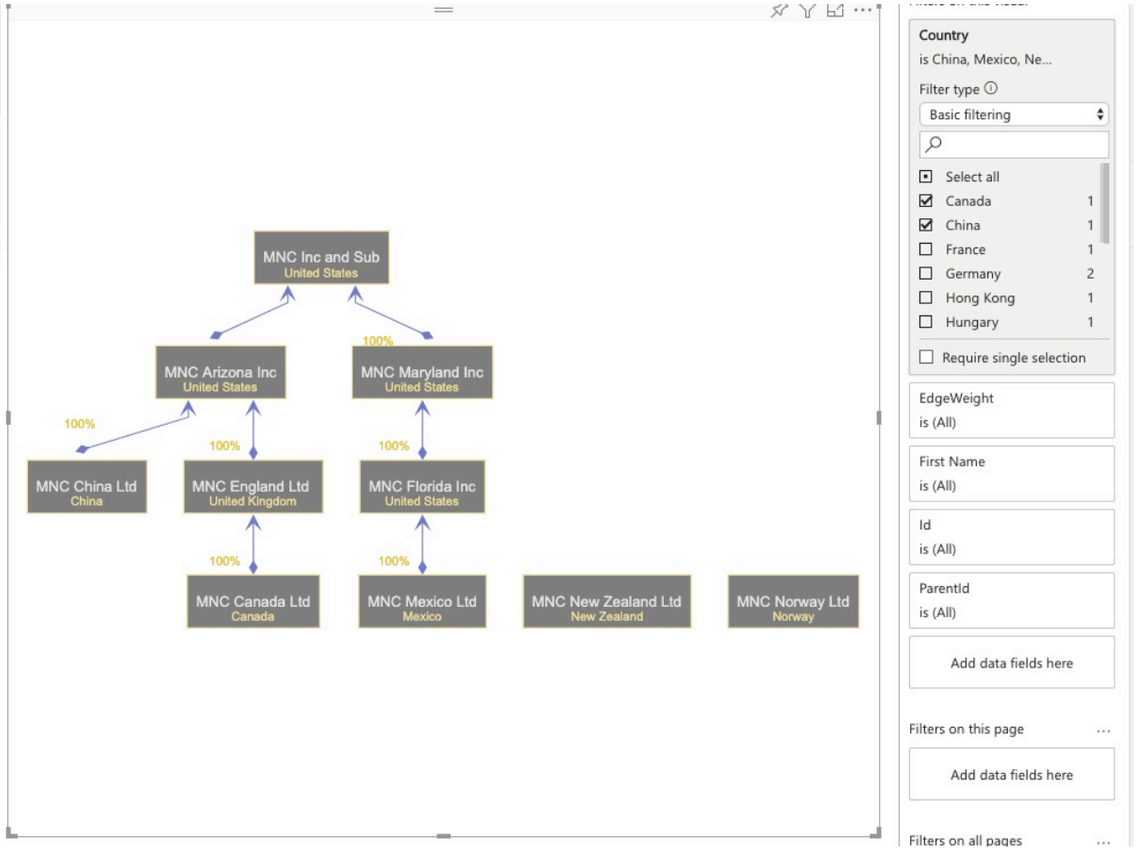
Styling

There are many style-oriented settings defined and they allow you to tune the look-and-feel of the diagram.

You can alter among other things:

- The node background and border
- The style of the main label as well as the secondary and top label
- The style of the edges and the label of the edge
- The usual style attributes defined by Power BI can be applied
- The layout algorithm applied
- The arrows on the edges.

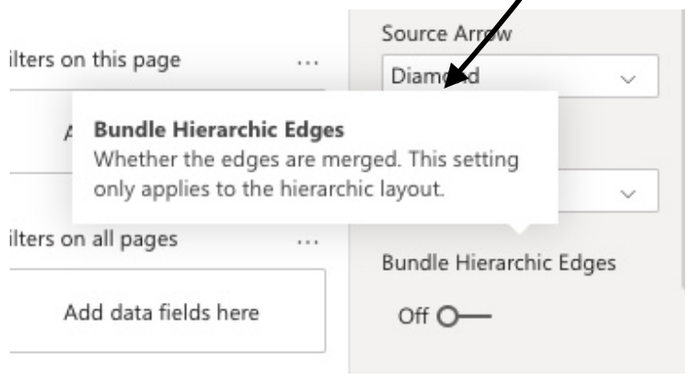
All of which can also help to brand the whole experience in function of customers and end-users.



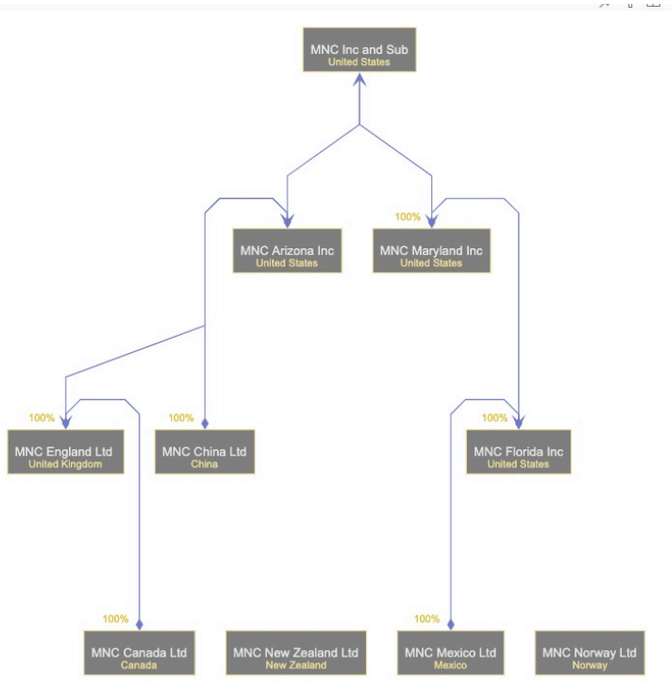
Filtering

Filtering of the diagram via the Filter panel in Power BI behaves as one would expect. The resulting graph can have disconnected elements but this is automatically handled by the layout.

Note in this respect an important setting for **the hierarchic layout which merges the edges**.



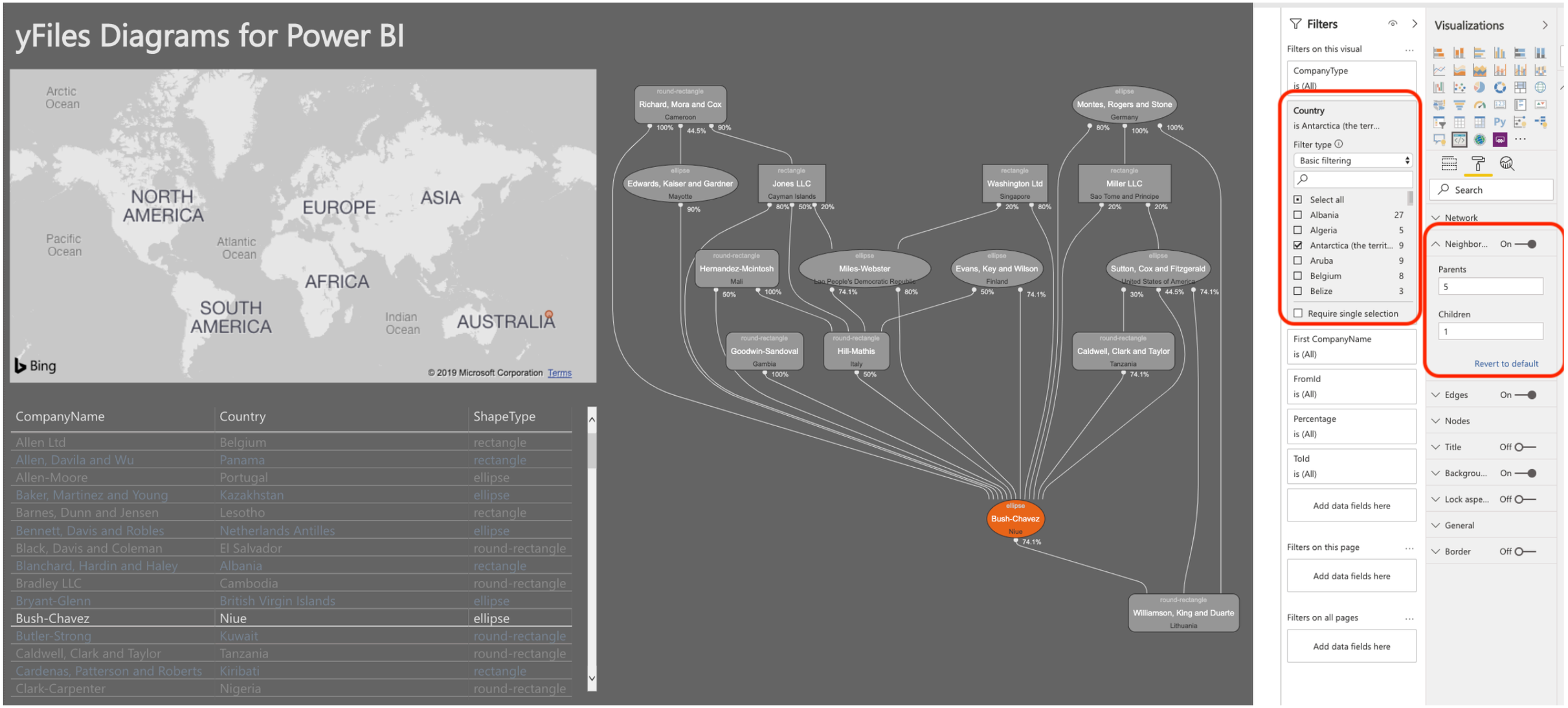
The first screenshot above shows the merging switched off while the screenshot below shows the merge switched on. The yFiles framework allow many variations and tweaks in this respect.



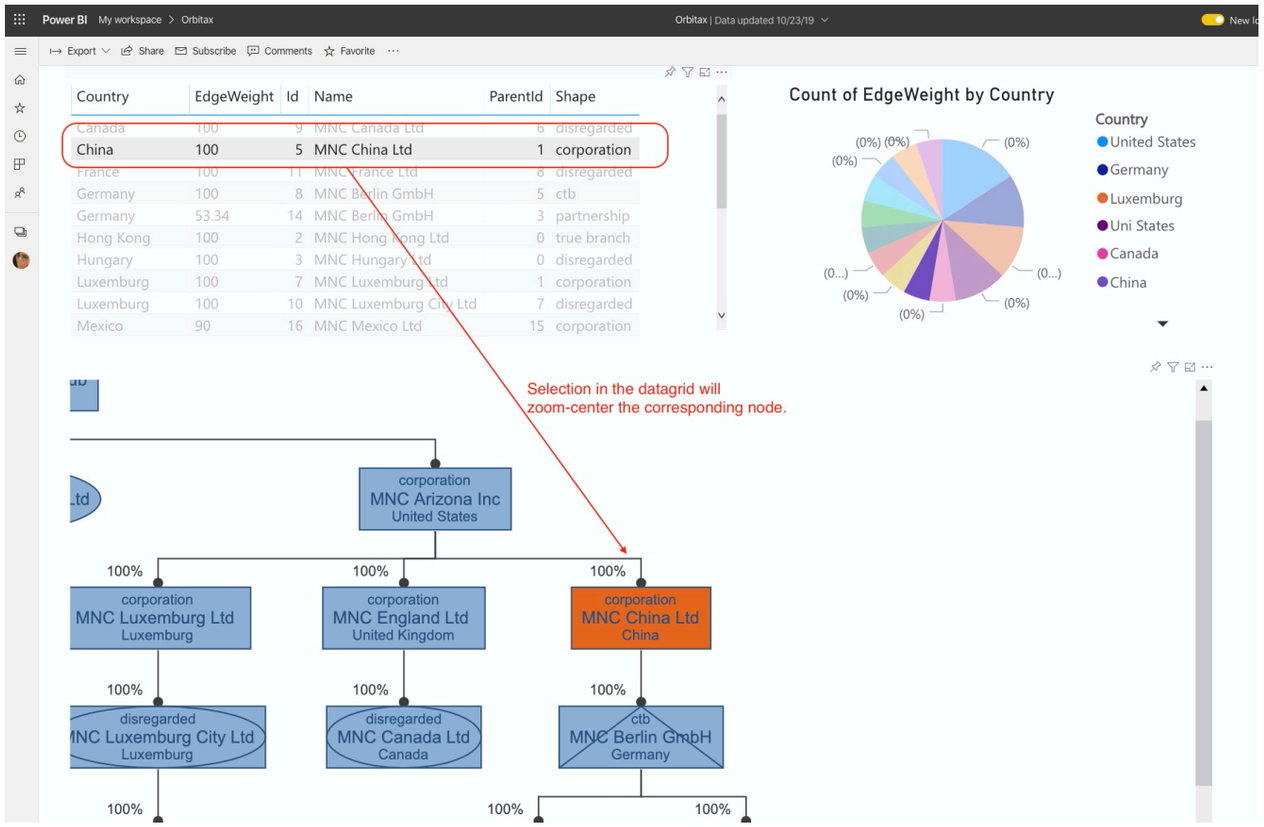
Neighborhood

Sometimes you want to focus on the vicinity of an item. The neighborhood setting allows you to filter the immediate parents and children of a node. The number corresponds to the depth in the hierarchy.

This neighborhood shows the whole diagram if no item is filtered or selected since the whole diagram is the neighborhood of itself.



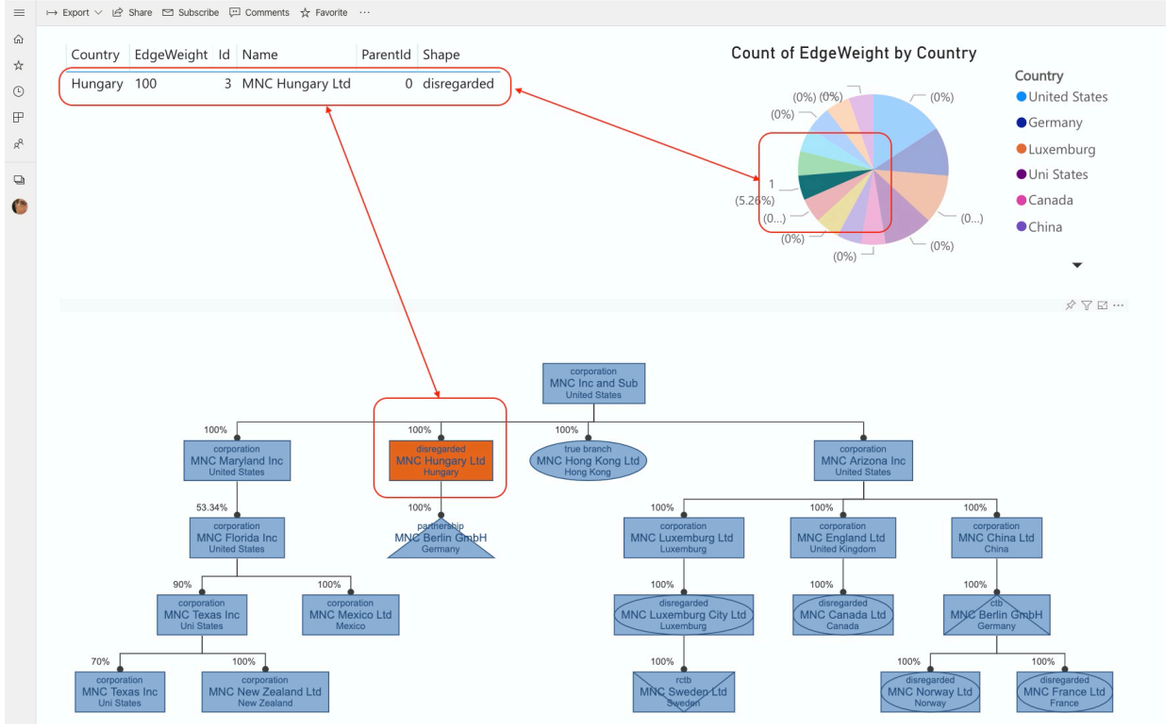
Slicing & Selection

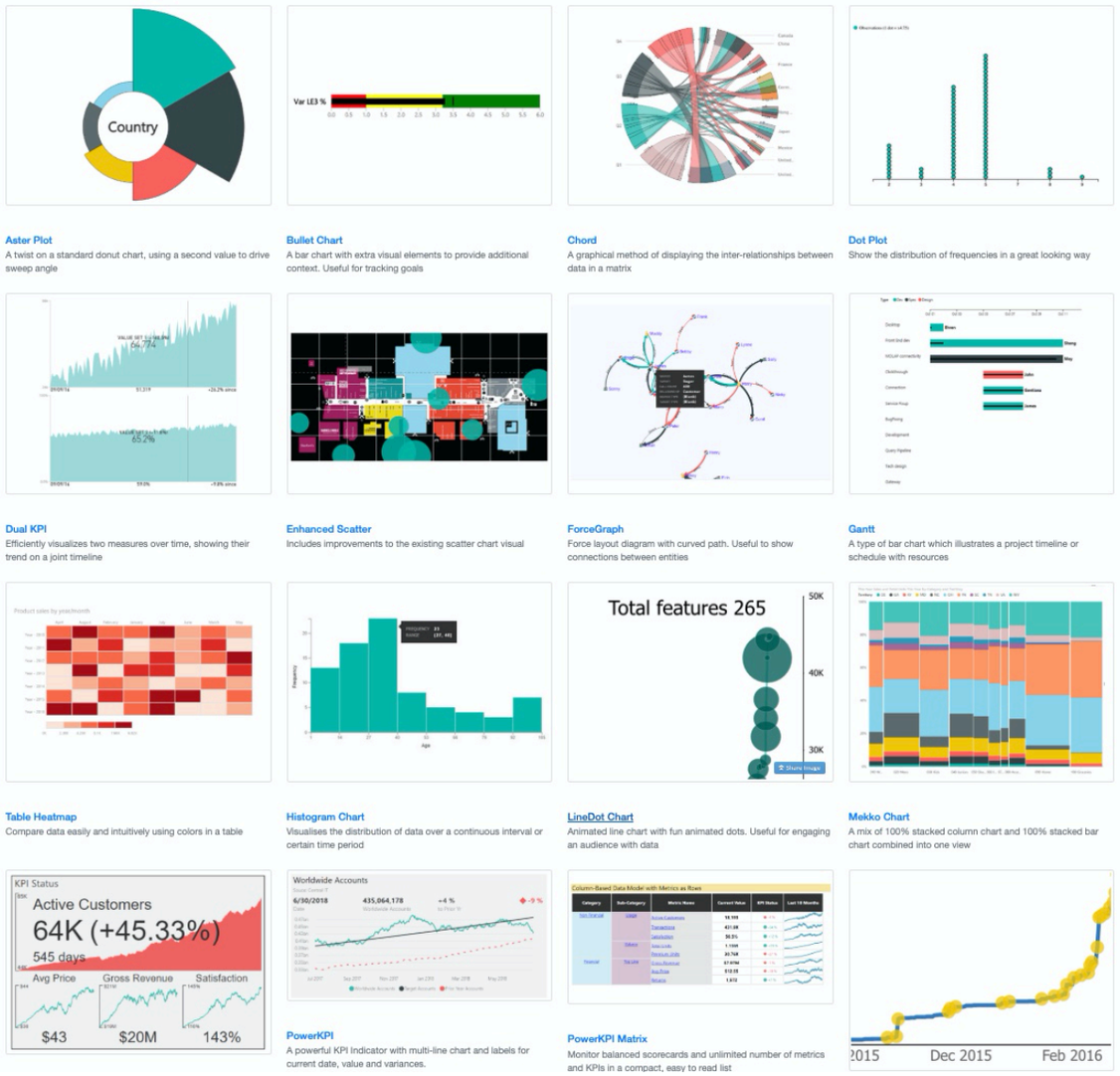


Nodes in the yFiles widget can be selected and it will trigger the same selection in adjacent report widgets (see screenshot). If the selection corresponds to multiple records in a widget the widget will likely show all the corresponding records. This is true for example for the datagrid but it's really up to the widget to decide how it handles the selection event.

Conversely, if you select an item in an adjacent widget the item will be highlighted and zoom-centered in the yFiles widget. In case there are multiple nodes possible the first one will be taken.

To deselect one should click on the empty canvas of the diagram. This will trigger a redraw of all the widgets. Deselecting all is similar in all widgets.





Power BI gives you a wealth of widgets to design business dashboards. The neatly work together with the yFiles widget via the standard Power BI mechanisms

About Custom Visuals

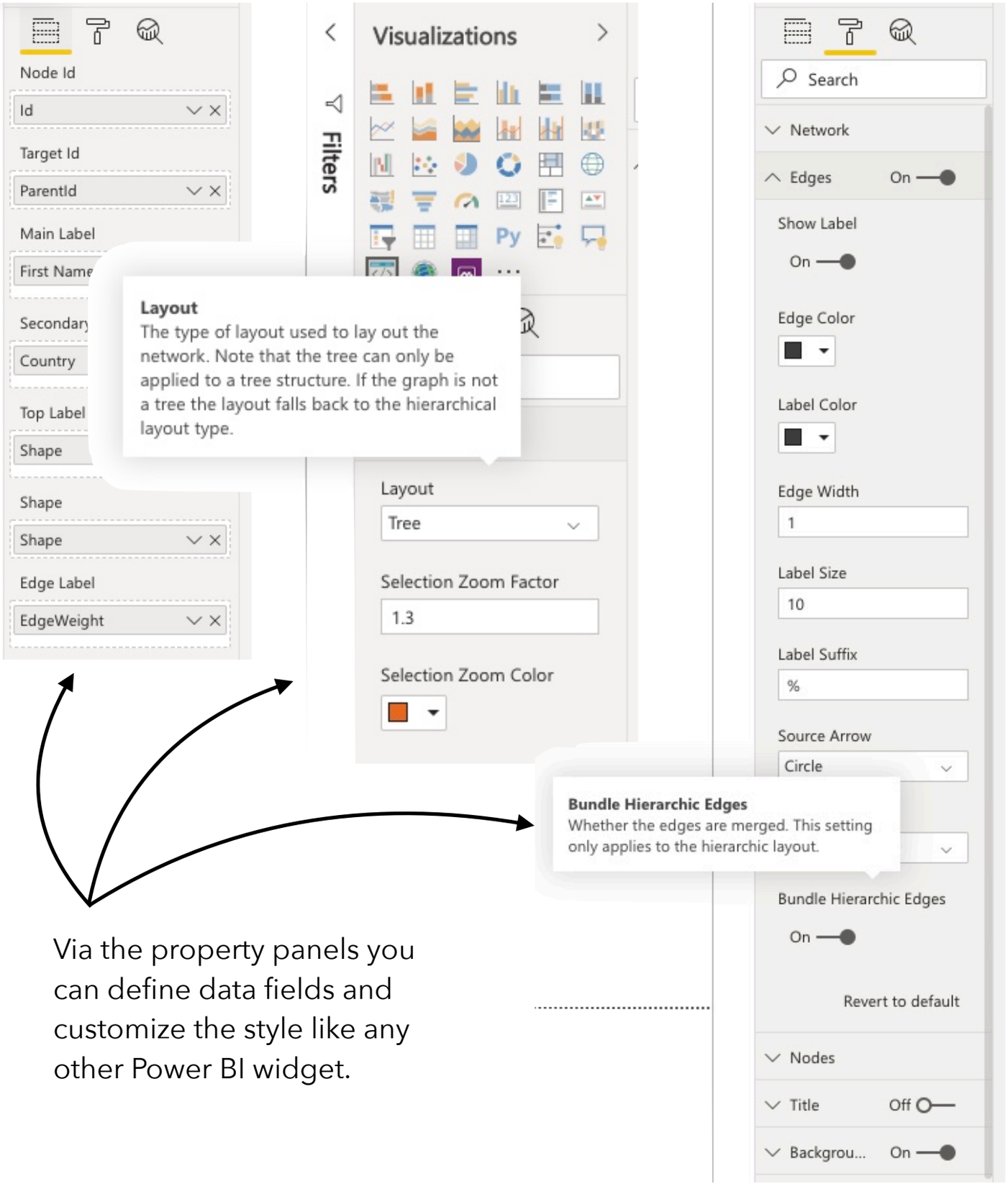
Developing a custom visual for Power BI is not very different from a standard HTML visualization. The ingredients are:

- a Power BI SDK giving various classes an utilities through which one can bind the inner working of the widget with the outer world where the data is defined, other widgets are reacting and so on.
- TypeScript
- A custom version of WebPack
- a CLI tool called pbiviz acting similarly to Angular.

Aside from this you are very free to do inside the widget whatever you like. The challenge is to use the appropriate API elements to react to focus, slicing, define custom properties and so on.

There are plenty of useful Github solutions showing how custom visuals operate. A neat overview is given here and it allows to click through to the actual source code on Github.

The absolute must-read article is the "Developing a Power BI visual" tutorial which highlights the essential steps and elements of a custom visual.



Via the property panels you can define data fields and customize the style like any other Power BI widget.

Defining Capabilities

The solution has a capabilities.json file which defines the properties of the widget in edit mode. The properties panel has three sub-panels:

- the data fields defining how the widget accesses the necessary values from a dataset
- the widget's properties: look-and-feel as well as behavior
- the analytical properties: not in use

The capabilities format is describe in this page. The names and values have to correspond to the settings.ts file. The json file can be seen as a serialization of the settings file. The settings.ts is crucial for the logic of the visual.

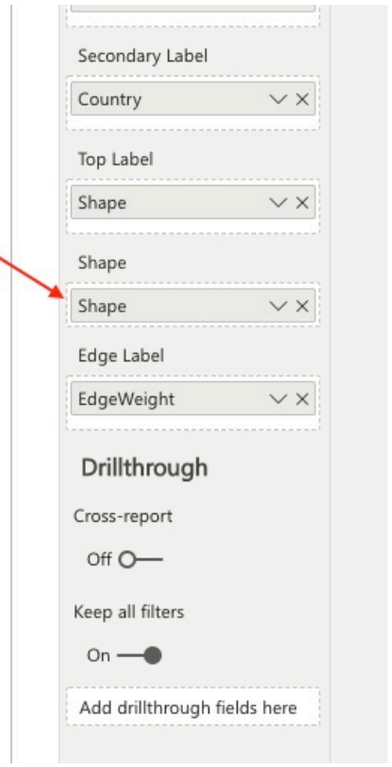
Note that the various elements of the property panels have a description which can be accessed by hovering over it (see adjacent screenshot).

Custom yFiles Shapes

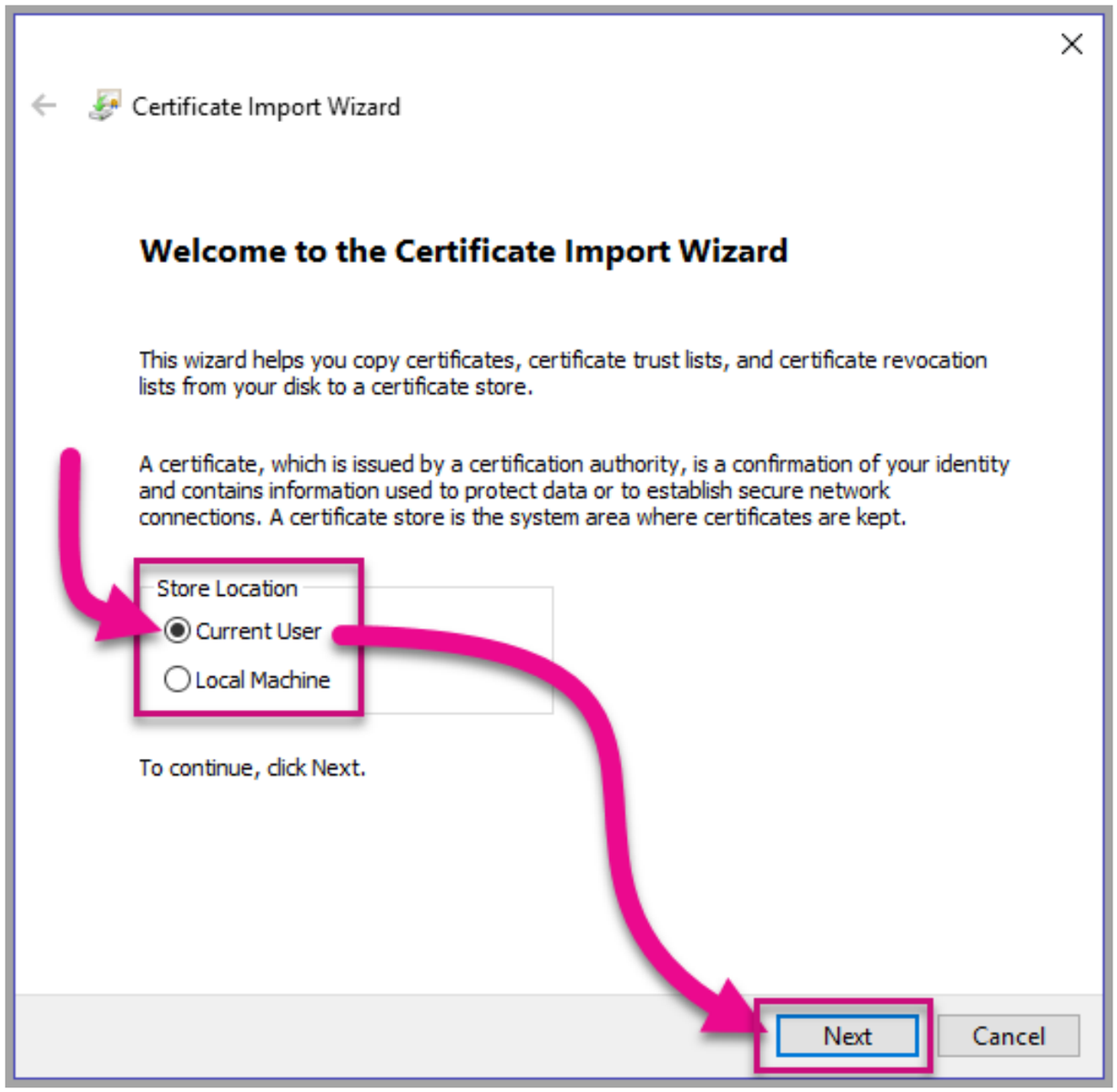
The shape of the nodes correspond to a field called `Shape` in the properties panel. The values are mapped to custom shapes in the diagram and if none provided the shapes will default to a rectangle.

The mapping corresponds to the given (business) names. It's easy to alter/extend this list and the custom yFiles visuals are also easy to add. It all amounts to a bit of SVG wrapped in an inherited `NodeStyleBase` class. The custom shapes can be found in the `shapes` directory of the solution.

Country	EdgeWeight	Id	Name	ParentId	Shape
Canada	100	9	MNC Canada Ltd	6	disregarded
China	100	5	MNC China Ltd	1	corporation
France	100	11	MNC France Ltd	8	disregarded
Germany	100	8	MNC Berlin GmbH	5	ctb
Germany	53.34	14	MNC Berlin GmbH	3	partnership
Hong Kong	100	2	MNC Hong Kong Ltd	0	true branch
Hungary	100	3	MNC Hungary Ltd	0	disregarded
Luxemburg	100	7	MNC Luxemburg Ltd	1	corporation
Luxemburg	100	10	MNC Luxemburg City Ltd	7	disregarded
Mexico	90	16	MNC Mexico Ltd	15	corporation




```
switch (name.toLowerCase()) {
    case 'true branch':
    case 'truebranch':
        return 'ellipse';
    case 'corporation':
        return 'rectangle';
    case 'partnership':
        return 'triangle';
    case 'disregarded':
        return 'reclipse';
    case 'rctb':
        return ‘envelope';
   case 'ctb':
        return 'inverted envelope';
    default:
        return 'rectangle';
  }
```



YOU DO NOT NEED TO INSTALL ANYTHING TO CONSUME OR CREATE A POWER BI DASHBOARD. THE TOOL AND THE CERTIFICATE ARE ONLY NECESSARY TO ALTER AND

The pbiviz Tool

The tool is crucial for the development of a Power BI widget. It can be installed like any other NodeJS tool:

```
npm i -g powerbi-visuals-tools
```

as a global package. The slightly more challenging part is that you also need to install a localhost certificate to make it work. More info can be found as part of this tutorial.

Setting up the certificate is necessary to allow an SSL communication between the localhost serving the custom visual and the online version of Power BI. It's important to emphasize also that

One cannot test and develop a widget with the desktop version.

The certificate is also a necessary requirement to compile the widget to a standalone package, as described below.

Compilation & Packaging

Using the `pbiviz` tool one can compile the custom visual to a `.pbiviz` file which can be imported into the toolbox. The same package works with the online version of Power BI and the desktop version. Simply execute:

```
pbiviz package
```

in the root of the solution and it will output in the `dist` directory the `.pbiviz` file.

A typical output looks like the following

```
>> pbiviz package

 info   Building visual...

 info   Start preparing plugin template

 info   Finish preparing plugin template

 info   Start packaging...

 info   Package compression enabled

 info   Package created!

 info   Finish packaging

 Webpack Bundle Analyzer saved report to ~/widget/
webpack.statistics.prod.html

 DONE  Compiled successfully in 2532350ms
```

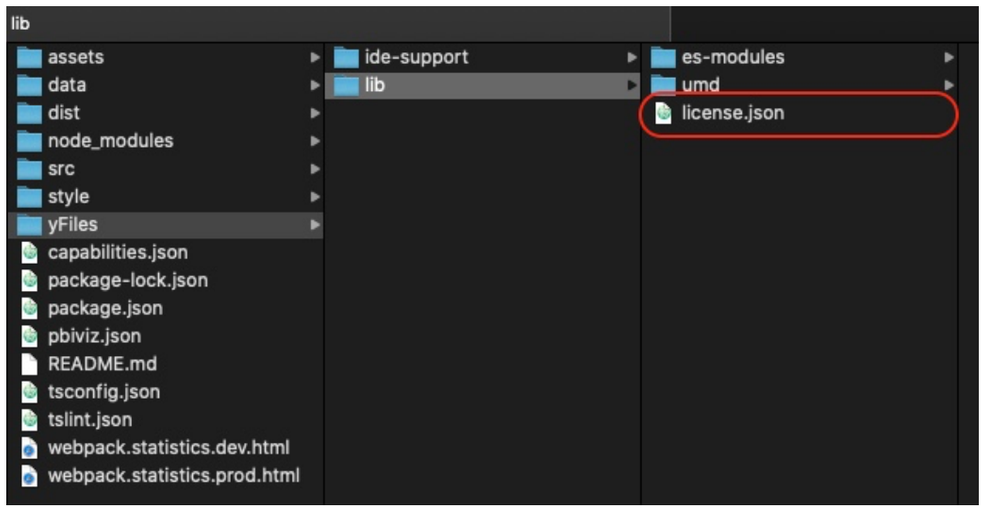
yFiles Dependency & License

The widget depends on the `lib` folder from the yFiles distribution. It has been included with the source code of the widget and is the latest release (v2.2.0.2).

The yFiles license sits also in this directory and is crucial, like any other yFiles application.

Of course, once the widget has been compiled as a standalone package (as explained below) the source code (including the license) becomes secondary. The widget is self-contained and works in both the desktop and online version of Power BI.

Obviously, like any other HTML application, you the need the source code to alter the widget and recompile it to a new version. This is in particular true if you wish one day to upgrade the underlying yFiles dependency.



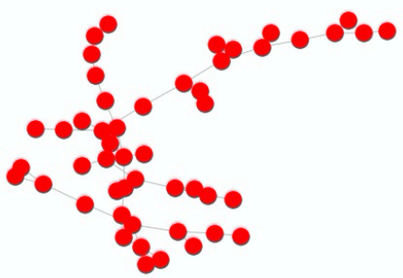
Orbitax data generator

This is a notebook with a simple script to generate random graph data one can use in the Power BI widget for testing purposes.

Erdos-Renyi random graph

```
[10]  g = nx.erdos_renyi_graph(50, 0.15)
      nx.draw(g, edge_color='silver')
```

Random tree



Export to CSV

```
[7]   import csv
      entityDic = {}

      def ensureEntity(id):
          if id not in entityDic:
                  companyType = faker.random.choice(["True Branch", "Corporation", "Partnership", "Disregarded", "CTB", "RCTB"])
                  entityDic[id] = {"country": faker.country(), "company": faker.company(), "type": companyType}
```

Random Testing Data

For testing purposes a Python utility script `generator.ipynb` was created and it has been included in the solution in the `data` directory. There is info in the document and should be opened as a Jupyter notebook. It allows to generate various pseudo-random graphs, including random trees and Erdos-Renyi graphs. The generated data can be exported as JSON or CSV and thereafter loaded in Power BI to test the yFile widget.

Note that the generated data also demonstrates the type of datasets the widget expects.

Although JSON data can be imported into the desktop version, one cannot do the same in the online version. Power BI desktop also articulates various data transformation interfaces one does not have in the online version.

On the other hand, the desktop version does not allow one to test/develop a custom visual. This can only be done via the online version.

yworks

yWorks GmbH
Vor dem Kreuzberg 28
72070 Tübingen
Germany
email: contact@yWorks.com

yWorks specializes in the development of professional software solutions that enable the clear visualization of graphs, diagrams, and networks. yWorks has brought together efficient data structures, complex algorithms, and advanced techniques that provide excellent user interaction on a multitude of target platforms. This allows the user to experience highly versatile and sophisticated diagram visualization in applications across many diverse areas.

Founded as a spin-off of the University of Tuebingen in 2001, yWorks today is the clear market leader, offering the best graph visualization and diagramming solutions available commercially.