Testing the `jupyter-viz` package

Nathan Carter, October 2018

Load the module

```
In [25]: LoadPackage( "jupyter-viz" );
```

```
Out[25]: true
```

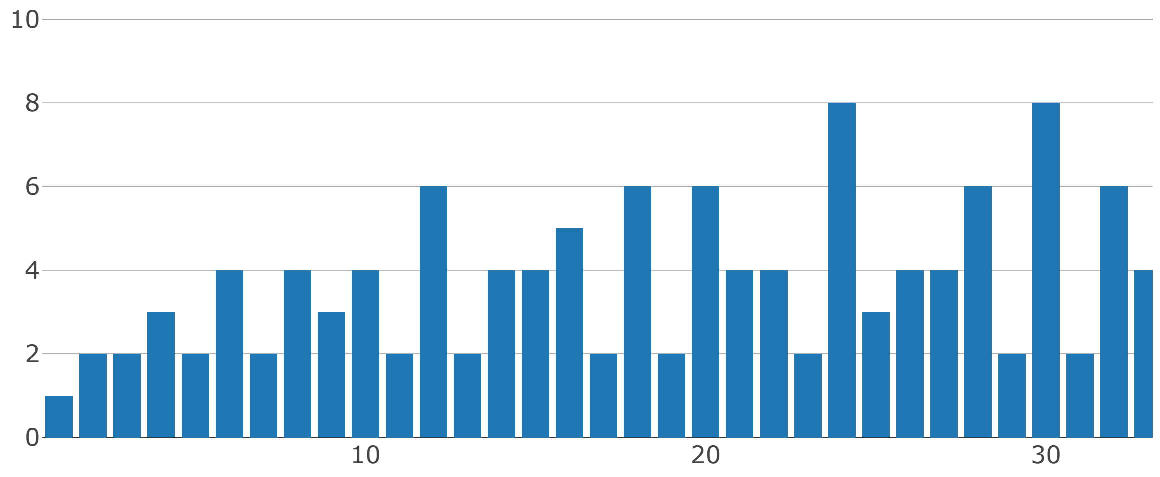
Test visualization with Plotly (https://plot.ly/)

For $n = 1$ to 50, how many divisors does n have?

Hover over the graph for popup information.

In [26]:
```
CreateVisualization( rec(
    tool := "plotly",
    data := rec(
        data := [
            rec(
                x := [1..50],
                y := List( [1..50], n -> Length(DivisorsInt(n)) ),
                type := "bar"
            )
        ],
        layout := rec( height := 400 )
    )
), "" );
```

Out[26]:



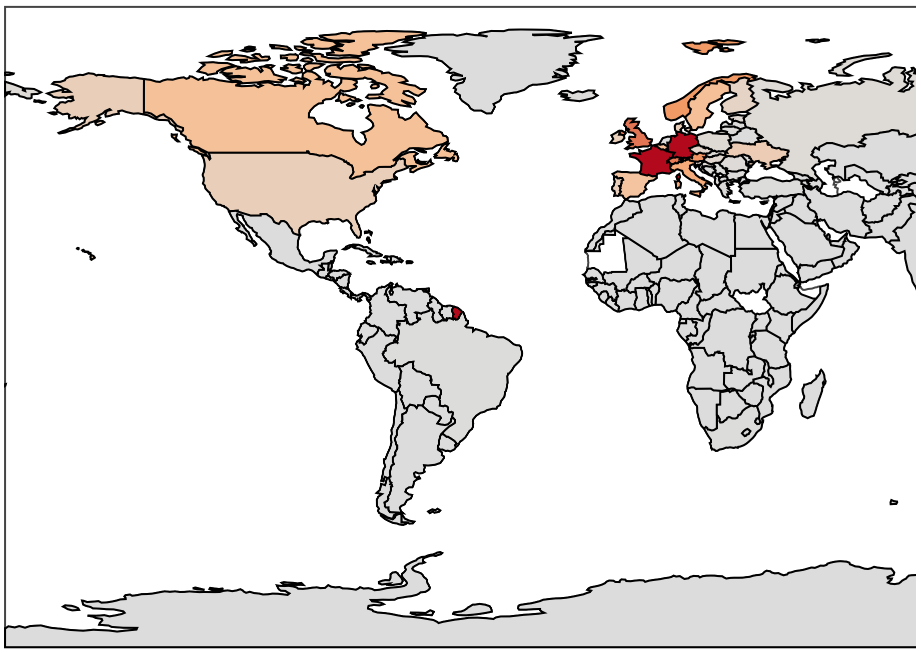
Load more complex Plotly chart from JSON file

This JSON file was downloaded from the Plotly gallery (https://plot.ly/~Dreamshot/9298), and contains data about number of electric vehicle charge points installed in 2017, worldwide.

Hover the graph for more information.

In [31]:
```
map := JsonStringToGap(
    ReadAll( InputTextFile( "EV Charge Points.json" ) ) );;
map.layout := rec( height := 500 );;
CreateVisualization( rec(
    tool := "plotly",
    data := map
), "" );
```

Out[31]:



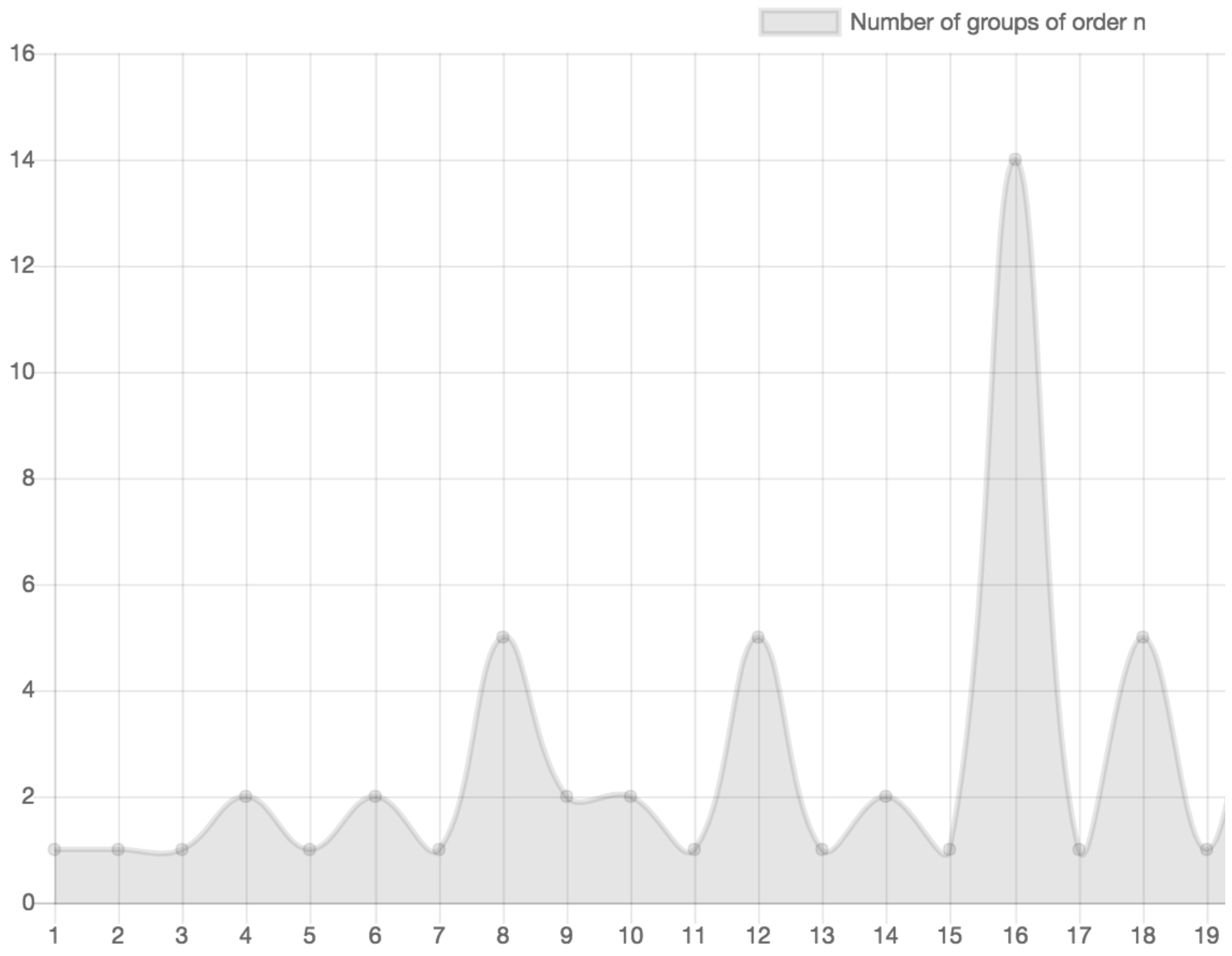
Test visualization with ChartJS (https://www.chartjs.org/)

For $n = 1$ to 30, how many groups are there of size n?

Hover the graph for more information.

In [32]:
```
CreateVisualization( rec(
    tool := "chartjs",
    data := rec(
        type := "line",
        data := rec(
            labels := [1..30],
            datasets := [
                rec(
                    label := "Number of groups of order n",
                    data := List( [1..30],
                        n -> Length( AllSmallGroups( Size, n ) )
                    )
                ),
            ]
        )
    )
), "" );
```

Out[32]:



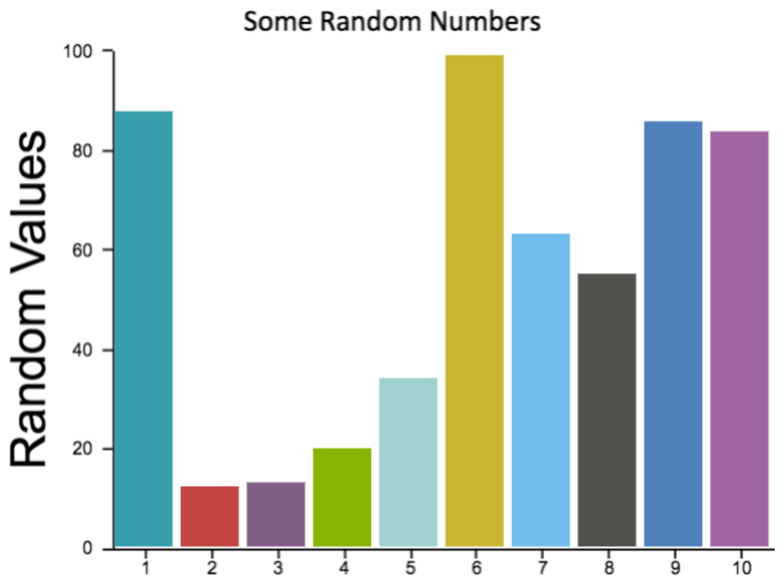
Test visualization with [CanvasJS (https://canvasjs.com/)](https://canvasjs.com/)

Just graphing 10 random integers in the range $\{1, \ldots, 100\}$.

Hover the graph for more "information."

In [34]:
```
CreateVisualization( rec(
    tool := "canvasjs",
    data := rec(
        animationEnabled := true,
        width := 400,
        height := 300,
        theme := "light2",
        title := rec( text := "Some Random Numbers" ),
        axisY := rec(
            title := "Random Values",
            titleFontSize := 24
        ),
        data := [
            rec(
                type := "column",
                dataPoints := List( [1..10],
                    n -> rec( x := n, y := Random( 0, 100 ) )
                )
            )
        ]
    )
), "" );
```

Out[34]:



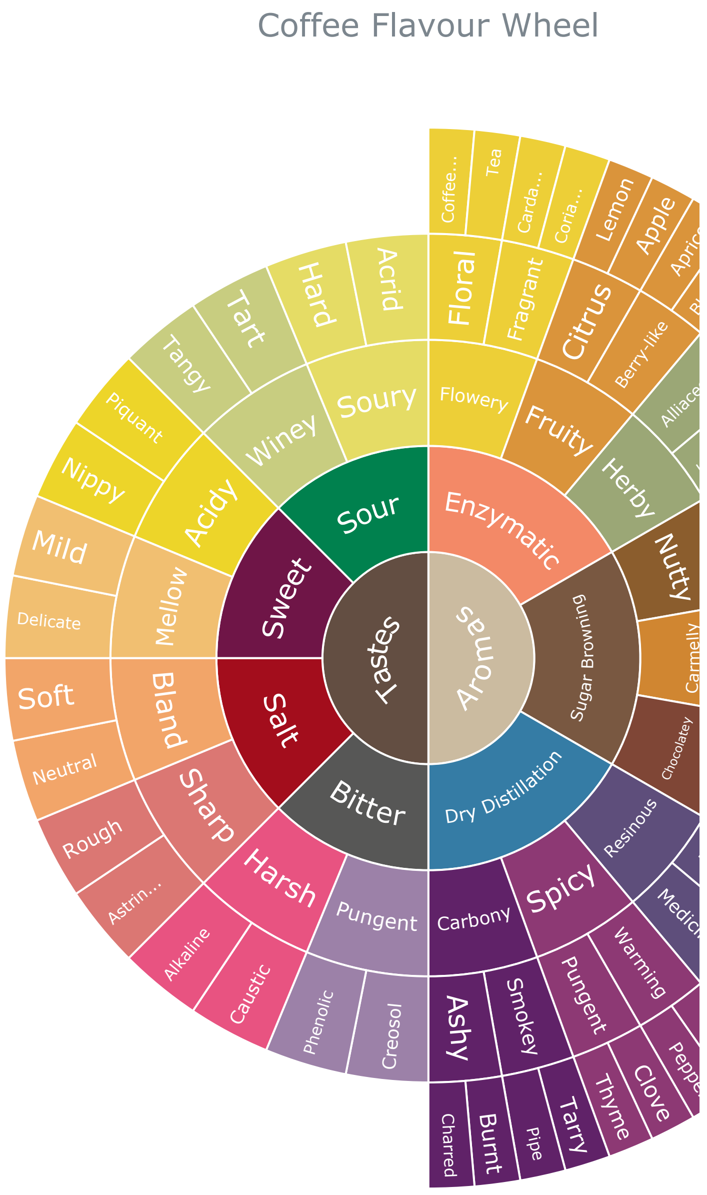
CanvasJS.com (http://canvasjs.com/)

Test visualization with AnyChart (https://www.anychart.com/)

This one was downloaded from the AnyChart gallery (https://www.anychart.com/products/anychart/gallery/) to show the flexibility of this toolkit, which is probably the most robust of all the ones shown on this page.

In [36]:
```
CreateVisualization( rec(
    tool := "anychart",
    data := JsonStringToGap(
        ReadAll( InputTextFile( "anychart-sample.json" ) ) )
), "" );
```

Out[36]:

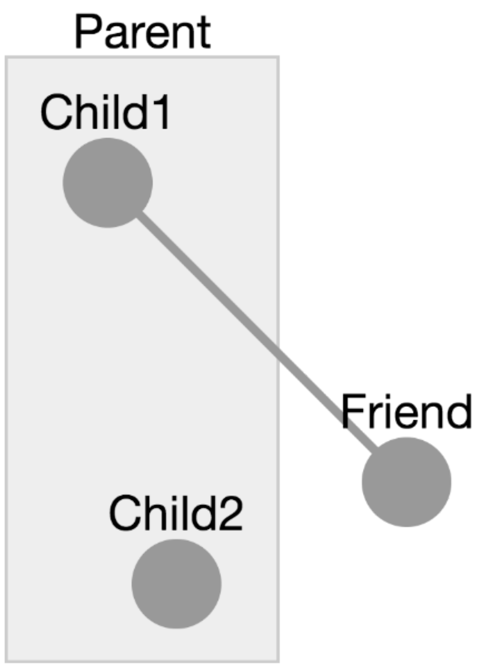


Test visualization with [Cytoscape (http://js.cytoscape.org/)](http://js.cytoscape.org/)

This simple graph was taken [from the Cytoscape documentation (http://js.cytoscape.org/#core/initialisation)](http://js.cytoscape.org/#core/initialisation) and slightly manipulated as part of this test.

In [38]:
```
CreateVisualization( rec(
    tool := "cytoscape",
    height := 400,
    data := rec(
        elements := [
            rec( # node 1
                group := "nodes",
                data := rec( id := "Child1", parent := "Parent" ),
                position := rec( x := 100, y := 100 ),
                selected := false,
                selectable := true,
                locked := false,
                grabbable := true
            ),
            rec( # node 2
                data := rec( id := "Friend" ),
                renderedPosition := rec( x := 200, y := 200 )
            ),
            rec( # node 3
                data := rec( id := "Child2", parent := "Parent" ),
                position := rec( x := 123, y := 234 )
            ),
            rec( # node parent
                data := rec( id := "Parent", position := rec( x := 200, y :=
            ),
            rec( # edge 1
                data := rec( id := "Edge1", source := "Child1", target := "
            )
        ],
        layout := rec( name := "preset" ),
        style := [
            rec( selector := "node", style := rec( content := "data(id)" )
        ]
    )
), "" );
```

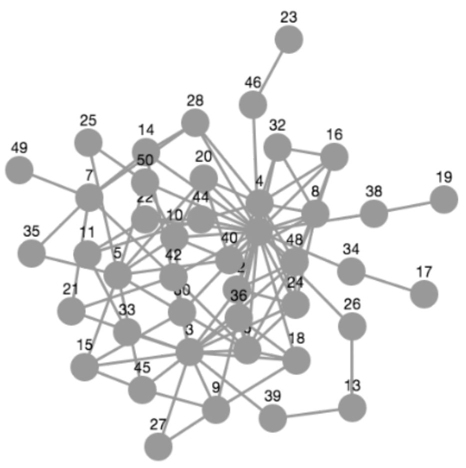
Out[38]:



Test creation of a graph with GAP code, then using Cytoscape for layout

```
In [43]: N := 50;;
         elements := [ ];;
         for i in [2..N] do
             Add( elements, rec( data := rec( id := String( i ) ) ) );
             if IsPrime( i ) then
                 Add( roots, i );
             fi;
             for j in [2..i-1] do
                 if i mod j = 0 then
                     Add( elements, rec( data := rec( source := String( j ), target
                 fi;
             od;
         od;
         CreateVisualization( rec(
             tool := "cytoscape",
             height := 600,
             data := rec(
                 elements := elements,
                 layout := rec( name := "cose" ),
                 style := [
                     rec( selector := "node", style := rec( content := "data(id)" )
                 ]
             )
         ), "" );
```

Out[43]:



```
In [ ]:
```