ChartML

Last Modified: 2008-12-03

ChartML is a xml document that can express one or more charts. ChartML document can be transformed to a SVG document with the XSLT file c2s.xsl. This document briefly explains the definition of ChartML.

Overview

The structure of the ChartML is the following.

- ViewArea (1)
  - Text (0..*)
  - Image (0..*)
  - ChartArea (1..*)
    - PlotArea (1..*)
      - Dataset (1..*)
      - Axes (1)
      - Grid (0..*)
      - Text (0..*)
      - Image (0..*)

The names of data types start with an upper case letter; the names of instances (elements) start with lower cases. The numbers in parenthesis represent cardinality of the elements in a document. For example, a document has one and only one ViewArea element, and the ViewArea element has one or more ChartArea elements.

Figure 1 demonstrates how the elements in the ChartML document are rendered in a web browser. The following sections explain each element in detail.

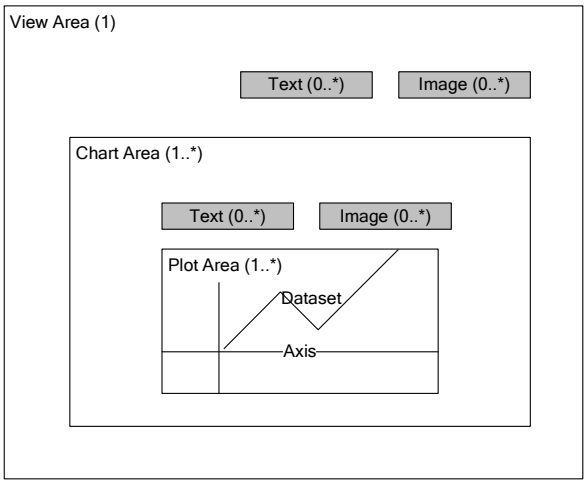

Figure 1 ChartML structure overview

ViewArea Element

ViewArea element is the root component of the document. It contains all chart related elements. A document must have one and only one ViewArea element.

ViewArea Coordinate System

The origin is at top left corner of ViewArea, and x value increases toward right, and y value increases toward down as shown in the below figure. The unit of major is in pixel. Figure 2 shows the coordinate system in View Area.

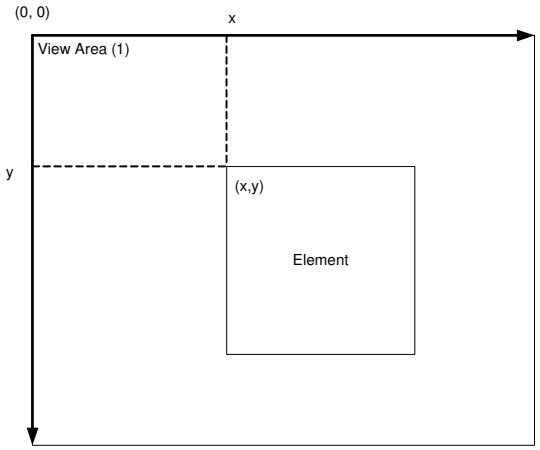

Figure 2 ViewArea coordinate system

ViewArea Attributes

The following are the main attributes of ViewArea.

width – width of the ViewArea.

height – height of the ViewArea.

class – class name that defines the appearance of ViewArea as CSS.

ViewArea Child Elements

The following are the elements that ViewArea can have.

text (0..*) – represents text(s) within the ViewArea. For example, it can be used to represent a title of the chart(s).

image (0..*) - represents image(s) within the ViewArea. For example, it can be used to represent a company logo for the chart(s).

charArea (1..*) – area(s) that used for chart.

Figure 3 demonstrates how elements in ViewArea are rendered in SVG document.

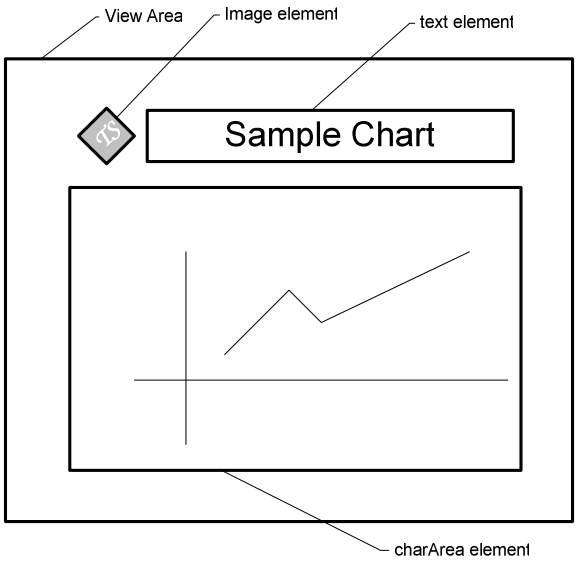

Figure 3 ViewArea output

ChartArea Element

ChartArea element represents a chart. A chart may be consists of title(s), legend(s), plot area(s), and label(s).

ChartArea Coordinate System

The origin is at top left corner of ChartArea, and x value increases toward right, and y value increases toward down as shown in Figure The unit of major is pixel. Figure 4 shows the coordinate system in ChartArea.

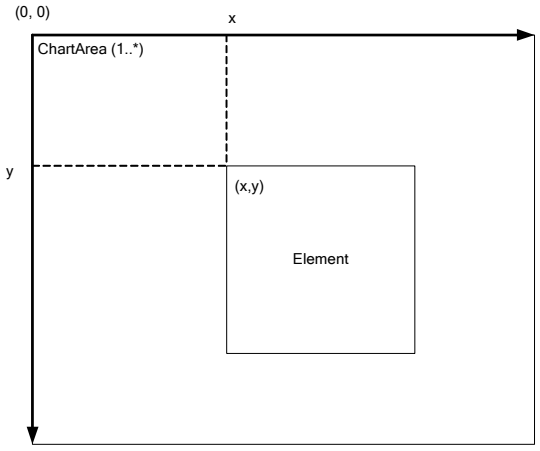

Figure 4 ChartArea coordinate system

ChartArea Attributes

The following are the main attributes of ChartArea.

x – x coordinate of the ChartArea in View Area coordinate system. Default value is 0.

y – y coordinate of the ChartArea in View Area coordinate system. Default value is 0.

width – width of the ChartArea. When not specified, the value is inherited from its parent, ViewArea.

height – height of the ChartArea. When not specified, the value is inherited from its parent, ViewArea.

class – class name that defines the appearance of ChartArea as CSS.

ChartArea Child Elements

The following are the elements that Chart Area can have.

text (0..*) – represents text(s) within the Chart Area. For example, it can be used to represent an axis title of the chart(s).

image (0..*) - represents image(s) within the Chart Area. For example, it can be used to represent a legend for the chart(s).

plotArea (1..*) – areas that draws chart from data.

Figure 5 demonstrates how elements are rendered in SVG document.

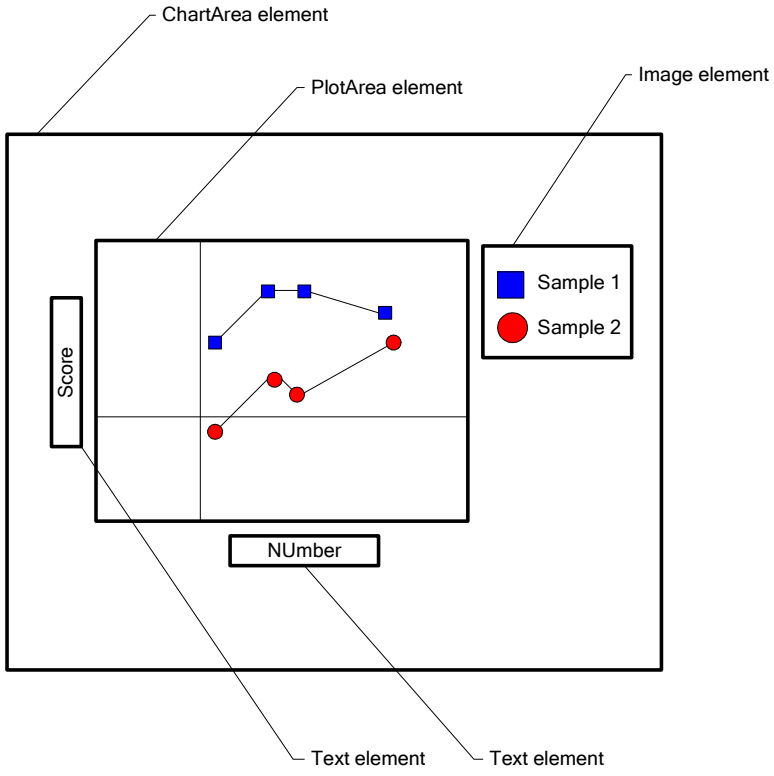

Figure 5 ChartArea output

PlotArea Element

PlotArea element represents a drawing part of chart. Currently a plot area can be line chart, scatter chart, column chart and bar chart. A plot contains axis, grid(s), data representation, (points, lines, or bars) and values on axis.

PlotArea Coordinate System

The coordinate system of PlotArea is defined by the axis. That means the origin of PlotAreas is the point of intersection of x axis and y axis. The unit of measure is the unit used for the axis and not pixel. Figure 6 demonstrate the coordinate system in PlotArea.

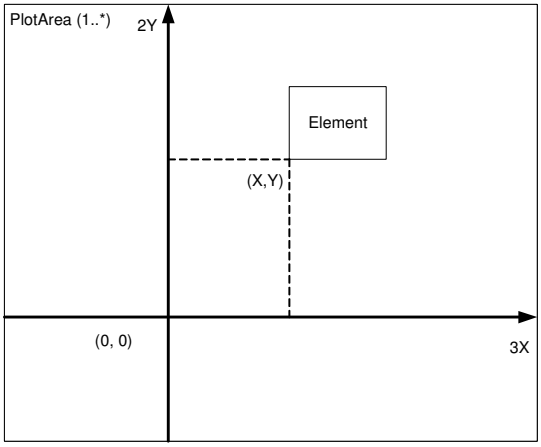

Figure 6 PlotArea coordinate system

PlotArea Attributes

The following are the main attributes of PlotArea element.

x – x coordinate of the PlotArea in ChartArea coordinate system. Default value is 0.

y – y coordinate of the PlotArea in ChartArea coordinate system. Default value is 0.

width – width of the PlotArea. When not specified, the value is inherited from its parent, ChartArea.

height – height of the PlotArea. When not specified, the value is inherited from its parent, ChartArea.

class – class name that defines the appearance of PlotArea as CSS.

PlotArea Child Elements

The following are the elements that PlotArea can have.

xAxis (1) – represents x-axe.

yAxis (1) – represents y-axe.

dataSet (1..*)– a set of data to be displayed as chart(s)

Figure 7 demonstrates how elements are rendered in SVG document for line chart PlotArea element.

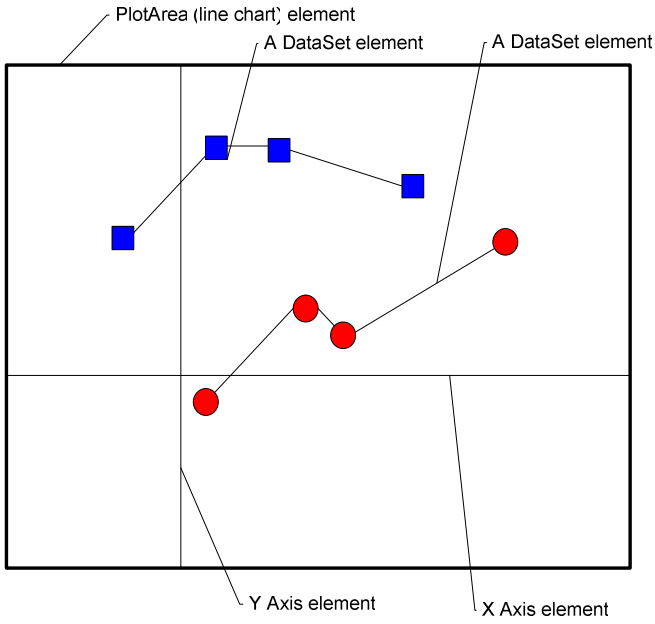

Figure 7 PlotArea output

Axis Element

Axis element represents an axis. There are two axis elements: x-axis and y-axis.

Axis Attributes

The following are the main attributes of Axis Element

leftValue (x axis only) – left value of the x axis.

rightValue (x axis only) – right value of the x axis.

topValue (y axis only) – top value of the y axis.

bottomValue (y axis only) – bottom value of the y axis.

class – class name that defines the appearance of axis line as CSS.

The width and height of the parent ChartArea and values in axis defines the scale of the chart plotting with the following formulas.

x-axis scale = (rightValue – leftValue) / width

y-axis scale = (topValue – bottomValue) /height

Axis Child Elements

The following are the elements that Axis element can have.

Grid (0..*)

Grid Element

Grid element represents grid line or/and tick marks. To make the schema simple, CharML doesn't provide an element explicitly for tick marks. Tick mark is represented as a grid line with a different style from a regular grid line. (Short and thick line usually.)

Grid Attribute

The following are the main attributes of Grid element

stepValue – distance between each grid in axis's unit. (Not pixel value.)

length – length of grid line in pixel.

x – x offset in pixel with which number value is displayed.

y – y offset in pixel with which number value is displayed.

numberFormat – XSL number format with which corresponding value is displayed. (Ex. **".0"** display number with one decimal point.)

class – class name that defines the appearance of grid line as CSS.

DataSet Element

DataSet element is a container for a set of data points to be displayed in a chart.

DataSet Attributes

The following are the main attributes of DataSet Element.

x – the x offset in the unit of pixel. With this attribute, multiple datasets can be displayed in one chart without over wrapping.

y – the y offset in unit of pixel.

class - The class name that defines the appearance of axis line as CSS.

7.2 DataSet Child Elements

Point (0..*) – a point in a dataset.

Point Element

Point element represents a datum in a dataset.

Point Attributes

The followings are the attributes for Point element.

xValue – x value of the datum with chart unit.

yValue – y value of the datum with chart unit.

size – size of the point to be displayed in pixel. It represents width of a bar for a column chart and a bar chart.

ChartML Example

Currently ChartML supports four types of charts: 1. Line chart, 2. Scatter (Bubble) chart, 3. Column chart, and 4. Bar chart. The following sections show a sample ChartML code with a CSS file for each chart type.

Line Chart

Figure 8 is an example ChartML document to create a line chart.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="xsl/c2s.xsl"?>
<chartML xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="xsd/ChartML.xsd">
<viewArea  width="800" height="500" >
  <text x="350" y="70" class="title">Sample Line Chart</text>
  <chartArea>
    <text x="350" y="480" class="label">X Axis Label</text>
    <text x="50" y="280" rotate="-90" class="label">Y Axis
Label</text>
    <image src="image/legend.png" x="650" y="150" width="140"
height="100" class="legend"/>
    <lineChart class="chart" x="130" y="120" width="500" height="300"
>
      <xAxis leftValue="0" rightValue="100">
        <grid stepValue="10" length="300" class="grid"   y="-20"
numberFormat=".0"></grid>
      </xAxis>
      <yAxis topValue="100" bottomValue="0">
        <grid stepValue="10" length="500" class="grid" x="-20" y="-5"
numberFormat="0"></grid>
      </yAxis>
      <dataSet>
        <point xValue="0" yValue="10"/>
        <point xValue="40" yValue="30"/>
                      :
        <point xValue="100" yValue="100"/>
      </dataSet>
      <dataSet class="line2">
        <point xValue="0" yValue="20"/>
                      :
        <point xValue="60" yValue="35"/>
        <point xValue="100" yValue="50"/>
      </dataSet>
    </lineChart>
  </chartArea>
</viewArea>
</chartML>
```

Figure 8 LineChart.xml

Figure 9 shows how LineChart.xml is represented in SVG document after applying c2s.xsl XSLT Stylesheet.

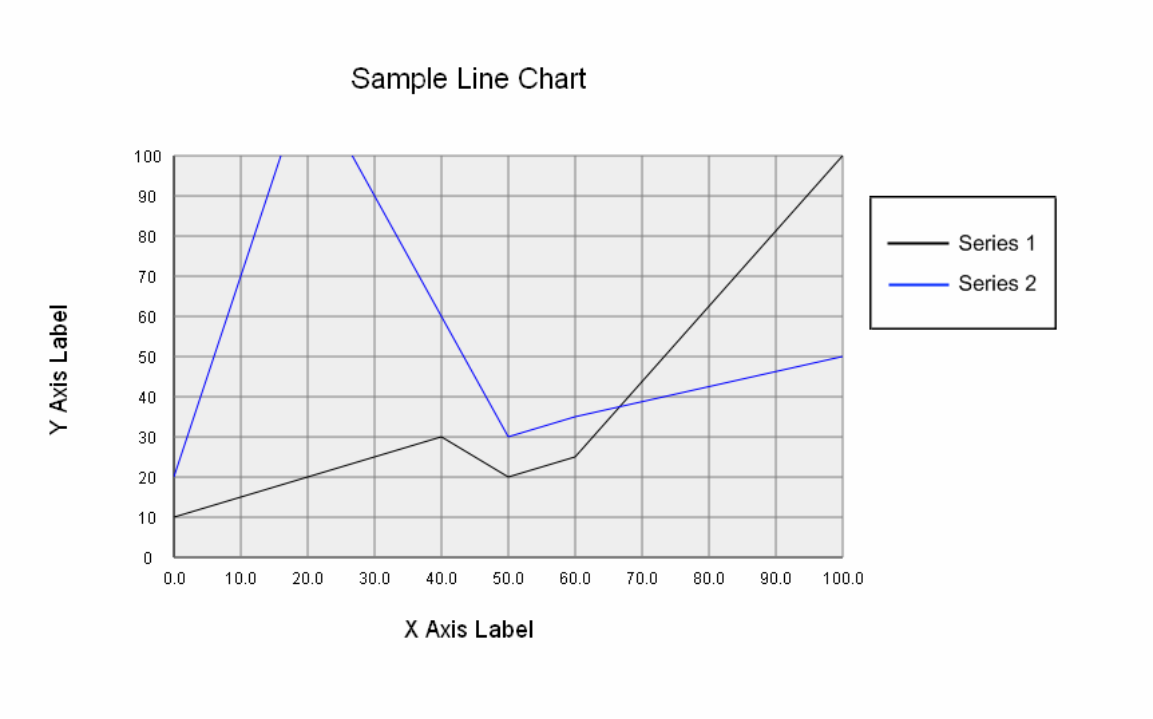

Figure 9 SVG representation of LineChart.xml

Please note that this example ChartML document uses style beyond the default setting (such as different color for series 2 line, etc.) And for this purpose, a CSS file is used.

Scatter/Bubble Chart

ChartML can express scatter chart and bubble chart with *scatterChart* element. Figure 10 is a ScatterChart element in BubbleChart.xml that represents a bubble chart.

```
    <scatterChart class="chart" x="130" y="120" width="500"
height="500" >
      <xAxis leftValue="-50" rightValue="50">
        <grid stepValue="10" length="250" class="grid" y="-20"
numberFormat=".0"></grid>
        <grid stepValue="10" length="-250" class="grid"></grid>
      </xAxis>
      <yAxis topValue="50" bottomValue="-50">
        <grid stepValue="10" length="250" class="grid" x="-20" y="-5"
numberFormat=".0"></grid>
        <grid stepValue="10" length="-250" class="grid"></grid>
      </yAxis>
      <dataSet class="point1">
        <point xValue="-45" yValue="-20" size="55"/>
        <point xValue="-20" yValue="5" size="30"/>
        <point xValue="0" yValue="5" size="20"/>
        <point xValue="35" yValue="40" size="10"/>
        <point xValue="47" yValue="42" size="5"/>

      </dataSet>
      <dataSet class="point2">
        <point xValue="-40" yValue="-20" size="35"/>
        <point xValue="-22" yValue="10" size="25"/>
        <point xValue="3" yValue="15" size="30"/>
        <point xValue="30" yValue="-49" size="50"/>
        <point xValue="47" yValue="40" size="8"/>
       </dataSet>
    </scatterChart>
```

Figure 10 ScatterChart element in BubbleChart.xml

Figure 11 shows the SVG representation of BubbleChart.xml document.

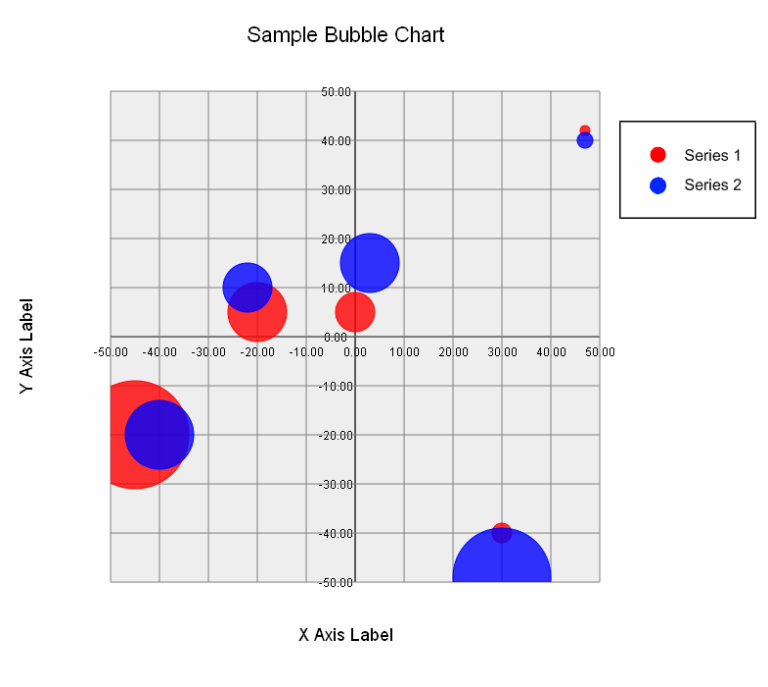

Figure 11 SVG representation of BubbleChart.xml

And CSS style in Figure 12 is used to improve the presentation of this bubble chart.

```
/* Bubble Style */
.point1 circle{
  fill: red;
 stroke: red;
  fill-opacity: 0.8;
}
.point2 circle{
  fill: blue;
  stroke: blue;
  fill-opacity: 0.8;
}
```

Figure 12 CSS for BubbleChart.xml

Column Chart

Figure 13 is a ColumnChart element in BubbleChart.xml that represents a column chart.

```
<columnChart class="chart" x="130" y="120" width="500" height="400" >
      <xAxis leftValue="0" rightValue="50">
        <grid stepValue="5" length="-5" class="grid"   y="-20"
numberFormat=".0"></grid>
      </xAxis>
      <yAxis topValue="50" bottomValue="-40">
        <grid stepValue="10" length="500" class="grid" x="-20" y="-5"
numberFormat="0"></grid>
      </yAxis>
      <dataSet class="col2">
        <point xValue="0" yValue="20" size="10"/>
        <point xValue="5" yValue="25" size="10"/>
        <point xValue="10" yValue="35" size="10"/>
        <point xValue="15" yValue="33" size="10"/>
        <point xValue="20" yValue="22" size="10"/>
        <point xValue="25" yValue="10" size="10"/>
        <point xValue="30" yValue="-15" size="10"/>
        <point xValue="35" yValue="13" size="10"/>
        <point xValue="40" yValue  ="10" size="10"/>
        <point xValue="45" yValue="15" size="10"/>
      </dataSet>
      <dataSet class="col1" x="7">
        <point xValue="0" yValue="30" size="10"/>
        <point xValue="5" yValue="15" size="10"/>
        <point xValue="10" yValue="49" size="10"/>
        <point xValue="15" yValue="13" size="10"/>
        <point xValue="20" yValue="12" size="10"/>
        <point xValue="25" yValue="-22" size="10"/>
        <point xValue="30" yValue="-35" size="10"/>
        <point xValue="35" yValue="-23" size="10"/>
        <point xValue="40" yValue="13" size="10"/>
        <point xValue="45" yValue="17" size="10"/>
       </dataSet>
</columnChart>
```

Figure 13 ColumChart element in ColumnChart.xml

Figure 14 shows the SVG representation of ColumnChart.xml document.

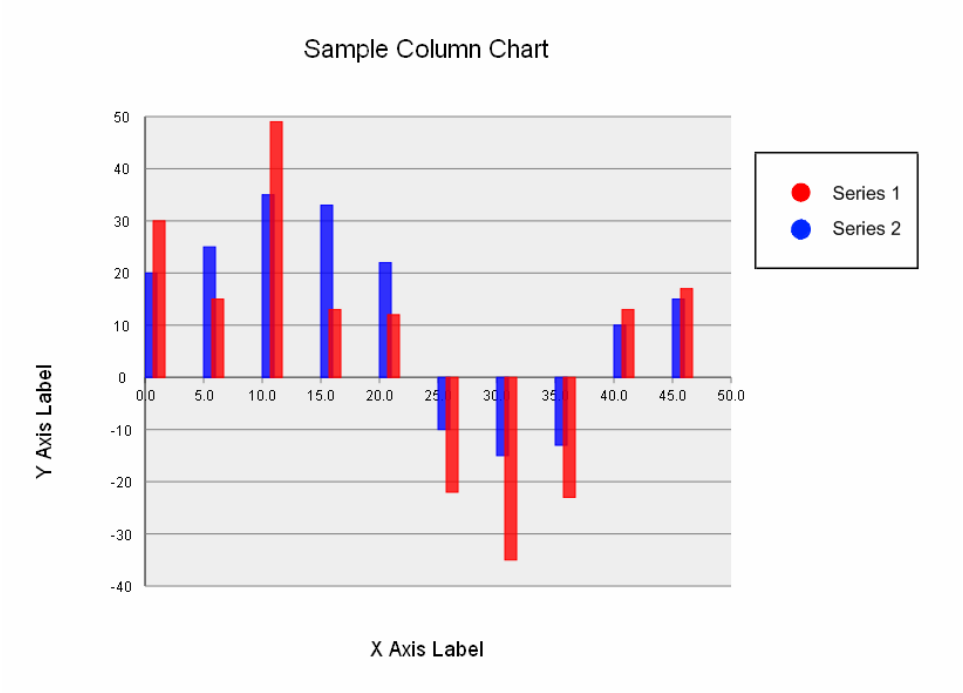

Figure 14 SVG representation of ColumnChart.xml

And CSS style in Figure 15 is used to improve the presentation of this column chart.

```
.col1 rect{
  fill: red;
 stroke: red;
  fill-opacity: 0.8;
}
.col2 rect{
  fill: blue;
  stroke: blue;
  fill-opacity: 0.8;
}
```

Figure 15 CSS for ColumnChart.xml

Bar Chart

Figure 16 is a BarChart element in BarChart.xml that represents a bar chart.

```
<barChart class="chart" x="130" y="120" width="500" height="350" >
  <xAxis leftValue="0" rightValue="50">
    <grid stepValue="5" length="5" class="grid" y="12"
numberFormat=".0"></grid></xAxis>
  <yAxis topValue="0" bottomValue="48">
    <grid stepValue="10" length="500" class="grid" x="-20" y="-5"
numberFormat="0"></grid></yAxis>
  <dataSet class="bar1" y="9">
        <point yValue="0" xValue="30" size="10"/>
        <point yValue="5" xValue="15" size="10"/>
        <point yValue="10" xValue="49" size="10"/>
        <point yValue="15" xValue  ="13" size="10"/>
        <point yValue="20" xValue="12" size="10"/>
        <point yValue="25" xValue="22" size="10"/>
        <point yValue="30" xValue="35" size="10"/>
        <point yValue="35" xValue="23" size="10"/>
        <point yValue="40" xValue  ="13" size="10"/>
        <point yValue="45" xValue="17" size="10"/>
      </dataSet>
  <dataSet class="bar2">
        <point yValue="0" xValue="20" size="10"/>
        <point yValue="5" xValue="25" size="10"/>
        <point yValue="10" xValue="35" size="10"/>
        <point yValue="15" xValue  ="33" size="10"/>
        <point yValue="20" xValue="22" size="10"/>
        <point yValue="25" xValue="10" size="10"/>
        <point yValue="30" xValue="15" size="10"/>
        <point yValue="35" xValue="13" size="10"/>
        <point yValue="40" xValue  ="10" size="10"/>
        <point yValue="45" xValue="15" size="10"/>
  </dataSet>
</barChart>
</chartML>
```

Figure 16 BarChart element in BarChart.xml

Figure 17 shows the SVG representation of BarChart.xml document.

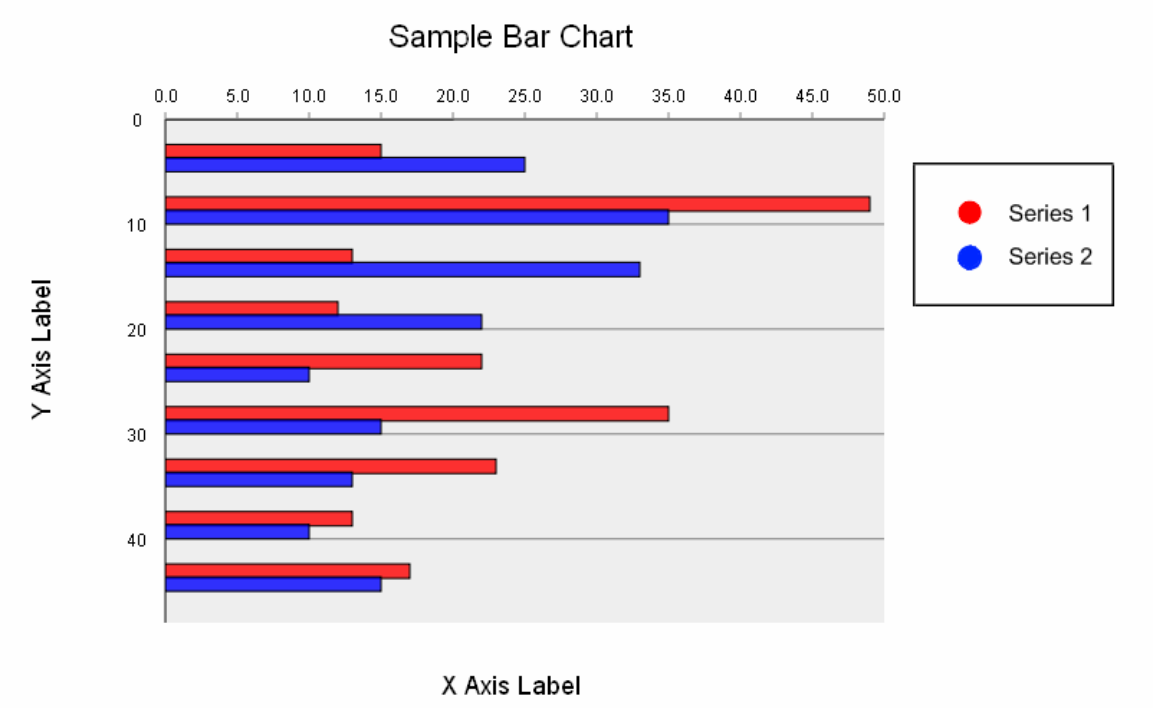

Figure 17 SVG representation of BarChart.xml

And CSS style in Figure 18 is used to improve the presentation of this bar chart.

```
.bar1 rect{
  fill: red;
 stroke: black;
  fill-opacity: 0.8;
}
.bar2 rect{
  fill: blue;
  stroke: black;
  fill-opacity: 0.8;
}
```

Figure 18 CSS for BarChart.xml