CDL Reference Guide

A Guide to the Chart Definition Language
Used in Visual Mining Products
Version 7.1
Summer 2011
## Table of Contents

1. **AN OVERVIEW OF CHART DEFINITION LANGUAGE (CDL)** .................................................. 4
   - Where Attributes Fit In ........................................................................................................ 6
   - CDL Statement Types ...................................................................................................... 7
   - Single Value ..................................................................................................................... 7
   - Tuple Values .................................................................................................................... 7

2. **VISUAL MINING’S SUITE OF CHART GENERATION SOLUTIONS** ........................................... 8
   - NetCharts Applets ........................................................................................................ 8
   - NetCharts Server ............................................................................................................. 8
   - NetCharts Pro .................................................................................................................. 9
   - NetCharts Designer ......................................................................................................... 9
   - Applets ........................................................................................................................... 10

3. **USING PARAMETERS IN CHARTING** ..................................................................................... 10
   - Standard HTML-Style Parameter Usage ........................................................................ 10
   - Using a Parameter Script to Specify CDL Parameters .................................................. 11
   - Parameters for Dynamic Charting ................................................................................. 11
     - Pointing to Parameters with a URL ............................................................................. 12
     - Putting Parameters in an Include File ......................................................................... 12
     - Using a Parameter Server .......................................................................................... 13
     - Getting Dynamic Updates from Another Applet ....................................................... 14
     - Dynamic Updates from JavaScript ............................................................................. 15
     - Loading Data Parameters ......................................................................................... 15

4. **TYPES OF AXES** .................................................................................................................. 17
   - Standard Axes ................................................................................................................ 19
   - Multi-Scaled Single Axes ............................................................................................... 19
   - Multiple Axes in a Single Direction ............................................................................... 19
   - Variable Axis Labeling .................................................................................................. 20
   - Logarithmic Axes .......................................................................................................... 20

5. **CDL PARAMETERS ARRANGED BY CHART TYPE** .......................................................... 21
   - Bar Chart and 3D Bar Chart .......................................................................................... 21
   - Box Chart ....................................................................................................................... 23
   - Bubble Chart .................................................................................................................. 24
   - Combo Chart ................................................................................................................... 25
   - Diagram/Map Chart ....................................................................................................... 27
   - Dial Chart ........................................................................................................................ 27
   - Heat Map .......................................................................................................................... 29
   - Histogram ....................................................................................................................... 30
   - Line Chart ....................................................................................................................... 31
   - MultiPie Chart .............................................................................................................. 33
   - Pareto Chart .................................................................................................................... 34
   - Pie Chart ......................................................................................................................... 35
   - Polar Chart ..................................................................................................................... 36
   - Radar Chart .................................................................................................................... 37
   - Stock Chart .................................................................................................................... 39
   - Strip Chart ..................................................................................................................... 40
   - Time Chart .................................................................................................................... 41
   - X-Y Chart ....................................................................................................................... 42

6. **CDL PARAMETERS ARRANGED BY FUNCTION** ................................................................. 44
ACTIVE LABELS AND DRILL-DOWN ................................................................. 44
  Parameters Involved, Alphabetically ....................................................... 44
AXIS MODIFICATIONS ................................................................................. 45
  Parameters Involved, Alphabetically ....................................................... 45
GRIDS ............................................................................................................. 48
  Parameters Involved, Alphabetically ....................................................... 49
LABELS ........................................................................................................... 49
  Font Names and Attributes ..................................................................... 49
  Parameters Involved, Alphabetically ....................................................... 49
LEGENDS ......................................................................................................... 50
  Parameters Involved, Alphabetically ....................................................... 50
NOTES, OR ANNOTATIONS ........................................................................... 51
  Parameters Involved, Alphabetically ....................................................... 51
REGIONS, OR BOXES ..................................................................................... 52
  Parameters Involved, Alphabetically ....................................................... 52
7. CDL PARAMETER DEFINITIONS ................................................................. 54

8. COMMON CDL ATTRIBUTES ..................................................................... 327

APPENDIX A: DATE AND TIME VALUES .................................................... 338
  DATE/TIME DATA INPUT .......................................................................... 338
    Mapping Date/Time Information .............................................................. 338
    Absolute Date Expressions ................................................................. 338
    Relative Time Units ............................................................................. 339
    Numeric Time Units ............................................................................ 340
  DATE/TIME DATA FORMAT AND TIMEBASE ........................................... 340
    SIMPLEDATE Format Expression Attributes ...................................... 341
    DATE Format Expression Attributes .................................................. 342
    TimeBase .............................................................................................. 342
    TimeUnit ............................................................................................... 343
1. An Overview of Chart Definition Language (CDL)

All of Visual Mining’s charting solutions use the Chart Definition Language (CDL) to create and manipulate charts. This common use of CDL makes it easier to recognize and preserve chart definitions when moving from one Visual Mining product to another. CDL is a simple ASCII scripting language that is easy to read and understand. CDL parameters live in a file with a file extension of .cdl or .cdx.

Visual Mining’s Charting solutions have many chart rendering details designed into the code. Each chart can be generated using very minimal CDL parameters without concern for details such as tic marks, font characteristics, grid lines, etc. As users become more proficient they can use CDL to create complex charts with a wide range of features and an informative and animated appearance.
1.1 CDL Statements

In the most general form, CDL parameter strings have the following form:

\[
\text{parameter1} = \text{value1}; \\
\text{parameter2} = \text{value2}; \\
\ldots
\]

Note: parameter strings can occur in any order and quotes can be single or double as long as they are matched.

The parameter names come from a defined set of CDL names, such as Background, Header, DataSet1. The value consists of one or more attributes.

Attributes are single primitive values that can be combined to form a complete value for a CDL parameter.

Example 1:
Background = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor);

In the above Example 1, the parameter Background has 6 attributes. Those attributes are: Color, BorderType, BorderWidth, ImageURL, ImageFormat and BorderColor. Note that attributes are separated by a comma.

In the below Example 2, we have assigned values to 4 of the attributes. The resulting chart will display with a white background, a black border which is raised using a width of 3.

Note: The trailing semicolon is required!

Example 2: Background = (white, RAISED, 3,,black);

CDL uses hundreds of attributes to describe colors, borders, width, depth, etc. For example, many parameters will use a color attribute to describe a color to be used in a chart.

The below attributes are Color Attributes:

<table>
<thead>
<tr>
<th>Bgcolor</th>
<th>LineColor</th>
<th>SymColor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fgcolor</td>
<td>BorderColor</td>
<td>TipColor</td>
</tr>
<tr>
<td>Color</td>
<td>FillColor</td>
<td>SliceColor</td>
</tr>
</tbody>
</table>
Where Attributes Fit In

The term “attributes” is somewhat context dependent, when we are speaking of CDL attributes. Generally, we mean the qualities or values assigned to some aspect or attribute of a chart, such as the names of the days of the week given as bar labels, or the string, “Weekday Network Load” given as the title of a chart.

CDL has a rich set of attributes—from 800 to 1000 altogether—which may be used in defining charts. The emphasis is on “may,” because all of the chart types possess defaults that produce a simple graph that is easily modified, and it’s not necessary to specify every little thing in order to get a decent result. The example below shows the CDL required to define a simple bar chart.

Example:

```cdl
ChartType = Barchart; <!-- type of chart -->
ChartWidth = 400; <!-- width of chart -->
ChartHeight = 250; <!-- height of chart -->
Header = ("Weekday Network Load"); <!-- title of chart -->
BottomTics = ("ON"); <!-- use tic marks on bottom -->
BarLabels = "Mon", "Tue", "Wed", "Thu", "Fri";
LeftTitle = ("Bytes\nPer\nSec"); <!-- add left axis title -->
DataSets = ("Server #1", blue); <!-- specify first data set -->
DataSet1 = 100, 125, 245.78, 147, 67; <!-- static data -->
```

Produced this chart, shown somewhat reduced in size:

![Figure 1: A Very Basic Bar Chart](image)

No actual programming—meaning, constructing logical algorithms in a programming language such as C++ or Java—was required to do this. The defaults construct a reasonable chart, so you don’t have to delve into the depths of CDL to make simple charts. You just need to know what parameters you want to define, and describe them. However, the depths of CDL are there, and very useful indeed for adding interactivity and expressive detail to your charts.

This CDL can be used in a variety of applications of visual mining products. It can be provided as a parameter to a NetCharts Applet. It can be loaded into a NetCharts Server, or provided to a NetCharts EJB. This CDL can also be modified and made even more robust by using ChartWorks Designer.
CDL Statement Types

CDL has 5 basic statement types, differing primarily in the number and type of attributes that make up the parameter value.

Single Value
A Single parameter value is one attribute.

Example 1: GraphType = STACKED;

In the above example, the value for parameter named GraphType has one attribute. In this case, the attribute is Type and the value is STACKED.

List Values
A List parameter value is a list of attribute values.

Example 1: DataSet1 = 100,202,340,500;
Example 2: BarLabels = "Qtr1","Qtr2","Qtr3","Qtr4";

Tuple Values
A tuple can be easily identified because tuples start and end with parentheses. A Tuple parameter value is a group of attribute values.

Example 1: Header = (“My Header”, black, Arial, 12, 90);

Tuple List Values
A Tuple List parameter value is a list of groups of attribute values.

Example 1: StockSet1 = (100,200,300),(102,234,490),(102,234,490);
Example 2: DataSets = (“cherries”,red), (“plums”,purple), (“apples”,red);

VTUPLE Values
A VTUPLE value is a tuple value that can contain a variable number of attributes.

DialTicLabels = (name, label1, label2,…, labelN);

The value for DialTicLabels is a Vtuple. The first attribute is the name of a dial, the rest of the attributes are the tic labels for that dial. If the chart does not need all the attributes in the Vtuple, it ignores them. If the chart needs more attributes than provided, it re-uses the last attribute.
1.2 Visual Mining’s Suite of Chart Generation Solutions.

Every product in the Visual Mining suite uses a set of Java based charting “engines.” Some of the products are designed for HTML authors and webmasters and do not require programming in order to generate a chart. Some products are designed for programmers to make use of CDL’s robust charting options and integrate them into their own applications. However, they all use the NetCharts Java charting applets whose features are assigned via CDL parameters. The following section describes Visual Mining products and how CDL is referenced in each product.

NetCharts Applets

NetCharts Applets is Visual Mining’s entry level chart generation product. It consists of a set of Java Applets that utilize the functionality of the Visual Mining charting engines to render charts. NetCharts is popular with users wanting to deploy applets in web browsers and users wanting to chart enable desktop applications written in Java. CDL parameters may either be used in the standard HTML format for applets, or within the NFParamScript format.

![Figure 2. The relationship of CDL to its chart](image)

NetCharts Server

- Is a service that can provide web applications with charts, tables, or complete HTML documents containing multiple charts and tables. It can be deployed by itself or used in conjunction with any existing web infrastructure, from the simplest CGI scripts to the most sophisticated enterprise application servers. NetCharts Server can interact with a variety of data sources to obtain the data to be presented in its charts and tables.

- Offers simple HTML form based building tools that allow users to define a data source, build charts and tables, and assemble those charts and tables into a report.

- Offers programming APIs and Tool Kits that allow developers to interact with the server using nearly any server-side web programming language such as ASP, JSP, Java, and CFML and Perl.

- Can be deployed as a platform for delivering Web Services.
• Is platform independent and can run in most Windows-based and Unix-based environments.

Figure 3. NetCharts Server combines charts and tables into dynamic formats.

**NetCharts Pro**

NetCharts Pro is a Java programmer-friendly chart generation solution. Its API allows it to be used in Integrated Development Environments such as IBM’s WebSphere Studio. NetCharts Pro can create images of charts in popular web formats such as JPEG and PNG, making it ideally suited for server-side use in chart enabling servlets or JSP pages.

**NetCharts Designer**

NetCharts Designer is a desktop visual design tool that can be used to create chart templates for use with NetCharts, NetCharts Pro, and NetCharts Server.
Applets

Java applications can be embedded within an HTML document by means of an applet. Applets are Java code intended to be run within the context of an applet viewer or by a web browser. An applet is placed within an HTML document by using the <APPLET> tag. The <APPLET> tag is a container for the Applet that allows for the definition and configuration of the applet. Space to display the Applet is reserved by using the WIDTH and HEIGHT attributes of the <APPLET> tag.

Just like regular Java applications, additional parameters can be passed into an applet. The <PARAM> tags are child elements of the <APPLET> tag and must be contained between the start and end tags of an <APPLET> tag. The primary means of configuring the properties of NetCharts is via parameter passing into the applet.

embeds a chunk of Java code in an HTML document by means of an applet. An applet occupies a rectangular area of a web page, even if it is only one pixel square. An applet may require that some initial values be set from the HTML document. One or more PARAM elements can be used to pass parameters to the applet before the applet starts running. PARAM elements go between the start and the end tags of an APPLET element.

Example:

```html
<APPLET CODE="simpleapp.class" NAME="myapp" WIDTH=100 HEIGHT=100>
  <PARAM NAME="bgcolor" VALUE="black">
  <PARAM NAME="fgcolor" VALUE="yellow">
</APPLET>
```

Populating an applet via parameters is the primary means of sending attributes into NetCharts. To make this requirement of using parameters within an applet tag easier and more efficient, we've modified the typical HTML of a series of parameter statements within the applet tag so that the names and values make sense in terms of the chart you are defining. This collection of parameter names and their attributes we call the Chart Definition Language, or CDL. Using CDL is fundamental to creating Visual Mining charts.

2. Using Parameters in Charting

There is more than one way to arrange parameters to create a chart with our applets. The first two ways, with standard applet usage and with a parameter script, will suffice for most web-based charting with static data. Going beyond this into the realm of dynamic charting and chart applications requires a bit more work. Beginners need only examine these first two ways of using parameters.

Standard HTML-Style Parameter Usage

The standard method for passing parameters to any Java applet is through the use of one or more <PARAM> tags, which are defined within the confines of the <APPLET> tag. Each <PARAM> tag is assigned a unique name and a value, which may be quoted. For example, the following HTML segment defines a pie chart that has a white background with a black shadow, a red header title, and three pie slices with specific values and labels:
Example

```html
<APPLET CODE=NFPiechartApp.class WIDTH=400 HEIGHT=400>
  <PARAM NAME=Background VALUE="(white, SHADOW)">
  <PARAM NAME=Header VALUE="('Piechart Demo', whitesmoke)">
  <PARAM NAME=Slices VALUE="(12, aliceblue,'Fred'),
                            (23, seagreen,'Sally'),
                            (15, salmon, 'Jim')">
</APPLET>
```

Notice that each parameter is defined within a separate `<PARAM>` tag and that the values can span multiple lines, provided that the value is a quoted string. Also note how the strings defined within the parameter value use single quotes to differentiate themselves from the entire value string.

## Using a Parameter Script to Specify CDL Parameters

The very first example, shown in Figure 1, used a parameter script. While the use of `<PARAM>` tags, as above, is common for a small number of parameters, for the sake of convenience and readability, the `NFParamScript` parameter may be used, where the assigned value can hold any number of CDL statements.

`NFParamScript` requires less redundant typing, and is easier to generate from within a CGI script. The following example defines the same pie chart as in Figure 4, although it uses a single `NFParamScript` in order to create the CDL.

Example:

```html
<PARAM NAME=NFParamScript VALUE ='
Background = (goldenrod, SHADOW);
Header  = ("Piechart Demo", whitesmoke);
Slices  = (12, aliceblue, "Fred"),
          (23, seagreen, "Sally"),
          (15, salmon, "Jim");
'>
</APPLET>
```

Notice how only a single `<PARAM>` tag is used, with a quoted, multi-line value definition, using tabs or spaces for readability. Also note how the single quote is used to delimit the `<PARAM>` tag value and the double quote is used to denote a string value as a parameter attribute.

It is important to note that each parameter definition in a `NFParamScript` is terminated within a semicolon.

If both individual `<PARAM>` tags and an `NFParamScript` are used to define parameters in the same applet, then the individual `<PARAM>` tags will be processed first. That is, the `NFParamScript` values will take precedence over the individual `<PARAM>` tag values, if the same parameter name is being defined.

## Parameters for Dynamic Charting

You can use applets for creating dynamic charts, if you need to:
- Change charts dynamically while they are being viewed
- Specify the chart characteristics at runtime
- Access a chart from multiple HTML pages
- Generate updates from another applet
- Load large data sets (>100 values) quickly

**Pointing to Parameters with a URL**

Instead of placing all of the parameter definitions within an HTML file, you can use a URL access and retrieve the parameter definitions. Generally, this URL would refer to a data file, but any URL can be used, including a CGI script or application that generates the parameters dynamically.

For example, you may have multiple HTML files that reference the same chart. In that case, maintenance is reduced if the chart definition is stored in a single file or generated dynamically by a single CGI script. In the latter case, the CGI URL could even be customized to generate a custom chart for a given HTML file.

Regardless of the form of the URL, the resulting content-type should be a `text/plain` or `application/x-cdl` sequence of parameter definitions. That is, the URL should be a text file containing NetCharts parameter definitions. For example, the following HTML segment defines a pie chart based on the parameters defined in the `piechart.dat` file on the anywhere.com web server.

**Example:**

```html
<APPLET CODE=NPiechartApp.class width=400 height=400>
  <PARAMETER NAME=NFParamURL VALUE="http://anywhere.com/piechart.dat">
</APPLET>
```

Note: When a relative URL is given, the URL is interpreted relative to the Document Base of the HTML file containing the applet.

The `NFParamURL` parameter is processed after any individual `<PARAM>` tags, but before any `NFParamScript`. This allows the URL to contain standard attributes that may be overridden by the local parameter definitions defined in the `NFParamScript`.

Browser security dictates that the URL cannot point to a Web server other than the one from which the HTML file was loaded.

**Putting Parameters in an Include File**

All NetCharts Applets support the use of the `IncludeFile` parameter to insert any CDL file into another CDL definition. The `IncludeFile` parameter has the following syntax:

```
IncludeFile = "urlpathname";
```

**Example:**

```
IncludeFile = "http://www.visualmining.com/demo/background.cdl";
IncludeFile = "../demo/background.cdl";
```
The filename given can be any URL that is valid for the environment in which the chart is being executed. For example, in a browser, if the chart is downloaded from a Web server, then the URL can specify a relative pathname or a full HTTP pathname.

An example of the IncludeFile parameter’s use is a company trying to enforce a standard Background style for all of its charts. You could use an include file to accomplish this. Replace the Background definitions in the CDL with an IncludeFile reference to the CDL file containing the company standard Background definitions.

Example:

File: Background.cdl
Background = (yellow, 3, RAISED);

File: MyChart.html
<APPLET NAME=mychart CODEBASE=/NetCharts
    CODE=NFBarchartApp.class WIDTH=400 HEIGHT=400>
    <PARAM NAME=NFPARAMSCRIPT VALUE='IncludeFile = "Background.cdl";
    DataSet1 = ...
    ...
    '>
</APPLET>

Note that if a relative pathname is specified, as in the example above, then the DocumentBase will be used as the start of the relative location.

Using a Parameter Server

In the same way that you specify a Web URL from which parameter definitions are read, you may specify an arbitrary TCP server from which definitions will be processed. In this latter case, however, the definitions can be processed throughout the applet's lifetime. That is, a parameter server can continuously update any or all of the chart parameters, providing for dynamic charting.

The value of the NFPARAMSERVER parameter has the following format:

name = NFPARAMSERVER value="hostname:port/arguments"

If NFPARAMSERVER is defined, it will be processed after all other parameter definitions have been processed. At that time, a connection is made to the given host and port, which is assumed to be a TCP server capable of generating parameter statements. The TCP server can be written using any language or utility desired. It need only generate a stream of text data that is equivalent to a NFPARAMSCRIPT.

Everything following the "/" in the parameter string will be passed to the parameter server upon connection, terminated with a new line character. This allows the server to determine the specific data required for this connection.

For example, the following HTML segment specifies a parameter server located at www.netcharts.com using port 2000. An initial line containing DataSet=Monday, User=Fred is sent to the parameter server on startup, presumably telling it which chart to generate.
Example:

```html
<APPLET code=NFPiechartApp.class width=400 HEIGHT=400>
<PARAM NAME=NFPParamServer
VALUE="www.netcharts.com:2000/DataSet=Monday, User=Fred">
</APPLET>
```

A Technical Note About Server Connection Processing

A background thread processes all parameter statements generated by the NFPParamServer while the chart is being displayed. An "Update" command can be sent at any time within the data stream to cause the chart display to be updated. That is, parameter definitions received from the parameter server are batched together and the chart is refreshed whenever an "Update" command is received.

While server connection commands are being processed a status message is displayed whenever a parsing error occurs in the input stream. This aids developers in determining when a server bug exists. After displaying the status message, the parser will flush the input stream to the next semicolon and attempt to continue processing.

A status message is also displayed if the server connection is broken prematurely, notifying the user of the broken connection. To properly close down a connection without displaying such a message, the Parameter Server should send the following command in the input stream:

```
CLOSE;
```

Note: The trailing semicolon is required!

Parameter servers may display arbitrary messages to the user at any time by using the STATUS command in the input stream, as follows:

```
STATUS "This is a status message";
```

When the STATUS command is processed, the message window will be displayed immediately.

Getting Dynamic Updates from Another Applet

Parameter definitions can be programmatically updated at any time throughout the life of a chart applet through the use of the loadParams() method. The loadParams() method accepts a string value, which consists of one or more parameter statements. As with the NFPParamServer, the loadParams() definitions are batched together and the chart is refreshed whenever an "Update" command is given.

In the following Java example, the current applet accesses a pie chart applet via the AppletContext() (a standard Java capability) and then executes the loadParams() method of the pie chart applet to update the slice values and labels, as well as the pie chart background color.

Example:

```java
AppletContext ac = getAppletContext();
NFPiechartApp pie = (NFPiechartApp)ac.getApplet("piechart");

if (pie == null) {
    System.out.println("Unable to access piechart");
} else {
    pie.loadParams("Background = (blue);
    +"Slices = 
    +" (12, 'Fred'),";
```
Dynamic Updates from JavaScript

The `loadParamsJS()` method of each NetCharts applet can be used to update chart parameters programmatically from JavaScript. The applet context is determined using the "document" object within JavaScript, as shown in the following example:

**Example:**
```javascript
var app = document.piechart;
app.loadParamsJS("Background = (blue);";)
app.loadParamsJS("Slices = (12,'Fred'),(23,'Sally'),(15,'Jim');")
app.loadParamsJS("Update;")
```

Loading Data Parameters

In general, the concatenation of String objects in Java is very slow, relatively speaking. When loading large data sets (i.e., more than 100 or so values), significant performance increases can be gained simply by using a `StringBuffer` object to create the expression and then submitting the `StringBuffer` itself to the `loadParams()` method. For example, this code…

```java
StringBuffer sb = new StringBuffer("BarSet1 = ");
for (int i=0; i<500; i++) {
    if (i > 0) sb.append (", ");
    sb.append (i);
}
sb.append (";");
bar.loadParams (sb);
```

…Is faster than the following code…

```java
String str = new String ("BarSet1 = ");
for (int i=0; i<500; i++) {
    if (i > 0) str += ", ";
    str += i;
}
str += ";";
bar.loadParams (str);
```

Java programmers can also define parameter data using raw values, instead of `String` or `StringBuffer` expressions, in order to gain some additional performance (for large data sets) or to streamline data management within the application or applet (small or large data sets.)

For example, the following code can be used to define a bar chart data set:

**Example:**
```java
Vector data = new Vector();
```
data.addElement (new Integer(27));
data.addElement (new Float(45.3));
data.addElement ("34");

bar.set ("BarSet1", data);

This eliminates the need to convert data vectors to comma separated strings in order to pass them to the loadParams() method.

Note: The Vector items can be defined using different object types, depending on the attribute type. In this case, the BarSet1 parameter expects a list of numbers, which can be defined using many different object types, including a String. The parser will automatically convert, if possible, items as needed when loading the data.

The specific object type used for the data depends on the type of the parameter definition. In the example above, the BarSet1 parameter accepts a vector of numbers, so the object type used is a Vector. For parameters that accept a single value, a raw value type is passed.
3. Anatomy of a Chart

Referring to the following illustrations may assist you in understanding the CDL parameters and attributes more fully.

The notes in blue, such as Grid, Bottom Scroll, and Axis Label, indicate notation that Visual Mining products use in creating charts. Titles in blue, such as Left Title, Footer, Header, Right Title, and Legend, which do not have blue lines pointing to them, are what those items are actually called in CDL.

The following items, which are annotated above, can appear on the top, bottom, left, or right sides of a chart (though we have omitted them for the sake of clarity in the figure, above) and they may be combined, as you desire:

- Axis Labels
- Axis Tics
- Titles
- Active Labels
- Scroll Bars
Thus, you could label and show tic marks all four axes, with multiple grids and backgrounds, if you so desired, and have four scroll bars as well.
4. Types of Axes

Standard Axes

An axis can have many functions. An axis provides a reference for measuring coordinates. An axis also provides a way for displaying tic marks and scales. NetCharts allows for the zero base line of an axis to be located at the Top, Bottom, Left and Right of the chart. Moreover, you can have a series of axes and a series of scale sets on one line.

Multi-Scaled Single Axes

CDL parameters for axes allow a single axis to have multiple scales. For example, the chart displayed below has a bottom axis with two scales. The number of scales an axis can have is theoretically unlimited although it will be practically bounded by the size of the chart.

CDL Example:

BottomTics = ("ON",black,"SansSerif",10,90,null);
LeftTics = ("ON",null,"SansSerif",10,0,null);
Footer = ("[Sundays shown in gray]",black,"SansSerif",10,0);
BottomFormat = (DATE,"%n %d","1/1/2001 12:01:00","1d");
# BottomTicLocations is an example of variable axis labeling.
# Tics are placed at the explicit locations specified.
BottomTicLocation = "1/1/2001 12:01:00","1/8/2001 12:01:00","1/15/2001 12:01:00","1/22/2001 12:01:00","1/29/2001 12:01:00","2/1/2001 12:01:00","2/2/2001 12:01:00","2/3/2001 12:01:00","2/4/2001 12:01:00","2/5/2001 12:01:00","2/6/2001 12:01:00";

BottomScaleSet = (-.5,31,7,80),(32,36.5,1,20);

Multiple Axes in a Single Direction

This feature allows multiple axes to be defined in the same physical space that is usually occupied by a single axis. In other words, one can define multiple left, top, bottom or right axes. Each axis can be independently defined and controlled. Up to 10 axes can be defined in each direction. The parameters that define each additional axis are the same as the standard axis parameters with the exception of a number at the end.
CDL example:
LeftFormat = (FLOAT,"$%dK",,);
LeftFormat2 = (FLOAT,"$%dK",,);
LeftFormat3 = (FLOAT,"$%dK",,);

The parameters LeftFormat and LeftFormat1 are considered the same for backward compatibility.

**Variable Axis Labeling**

This feature allows additional control over axis tic mark drawing and labeling. The drawing of tic marks on each axis can be specifically enabled or disabled.

There are 8 CDL parameters for variable tic labeling, one for each axis location.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TopTicLocations</td>
<td>Top Axis Tic Mark Locations</td>
</tr>
<tr>
<td>BottomTicLocations</td>
<td>Bottom Axis Tic Mark Locations</td>
</tr>
<tr>
<td>LeftTicLocations</td>
<td>Left Axis Tic Mark Locations</td>
</tr>
<tr>
<td>RightTicLocations</td>
<td>Right Axis Tic Mark Locations</td>
</tr>
<tr>
<td>TopTicLength</td>
<td>Top Axis Tic Mark Length</td>
</tr>
<tr>
<td>BottomTicLength</td>
<td>Bottom Axis Tic Mark Length</td>
</tr>
<tr>
<td>LeftTicLength</td>
<td>Left Axis Tic Mark Length</td>
</tr>
<tr>
<td>RightTicLength</td>
<td>Right Axis Tic Mark Length</td>
</tr>
</tbody>
</table>

**Logarithmic Axes**

NetCharts 4.0 supports logarithmic x and y axes.

```plaintext
ChartName = "Logarithmic Y Scale";
ChartType = XYCHART; ChartWidth = 400;
ChartHeight = 300;
Background = (white,NONE,0,null,TILE,black);
Header = ("Base 16 Logarithmic Y-Scale",black,"SansSerif",14,0);
HeaderBox = (null,NONE,1,null,TILE,black);
BottomTics = (ON,black,"Courier New",10,null);
BottomScale = ("1200","2600",);
LeftTics = (ON,black,"Courier New",10,0);
LeftScale = ("300","6000","1000");
LeftScaleMode = (LOG,16);
```

Using LOG requires that the scale minimum be non-zero;
5. CDL Parameters Arranged by Chart Type

Use this index as a way of identifying parameter definitions to use when you have a specific chart to assemble. Chart styles are arranged alphabetically.

**Bar Chart and 3DBar Chart**

*Generally Supported Parameter Types*

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveLabels[1-50]</td>
<td>ActiveLabels[1-50] = (&quot;Label1&quot;, &quot;URL1&quot;, &quot;Target1&quot;), ...</td>
</tr>
<tr>
<td>Axis</td>
<td>See also Chapter 6, Active Labels and Drill-Down</td>
</tr>
<tr>
<td>Axis Thickness</td>
<td>AxisThickness = 15;</td>
</tr>
<tr>
<td>Color Table</td>
<td>See Chapter 6, Axis Modifications for various parameters available</td>
</tr>
<tr>
<td>BuildAnimationEnabled</td>
<td>BuildAnimationEnabled = ON</td>
</tr>
<tr>
<td>Background</td>
<td>Background = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor);</td>
</tr>
<tr>
<td>BackgroundFillPattern</td>
<td>BackgroundFillPattern = (type, color1, color2, imageURL), ...;</td>
</tr>
<tr>
<td>ChartElementSpacing</td>
<td>ChartElementSpacing = spacing;</td>
</tr>
<tr>
<td>Grid</td>
<td>Grid = (LineColor1, bgColor1, borderColor1, bgImage1, ImageFormat1), ...;</td>
</tr>
<tr>
<td>Legend</td>
<td>Legend = (&quot;Label&quot;, Color, &quot;FontName&quot;, FontSize, Angle, interiorAlignment);</td>
</tr>
<tr>
<td>Note Sets</td>
<td>NoteSets = (&quot;Name1&quot;, Justify1), (&quot;Name2&quot;, Justify2), ...;</td>
</tr>
<tr>
<td>Title</td>
<td>See also Chapter 6, Notes, or Annotations</td>
</tr>
<tr>
<td>TitleBox</td>
<td>See Chapter 6, Labels for various parameters available.</td>
</tr>
</tbody>
</table>

*Specifically Supported Parameters*

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BarAnimationStyle</td>
<td>BarAnimationStyle = GROW</td>
</tr>
<tr>
<td>Bar3DDepth</td>
<td>Bar3DDepth = Number;</td>
</tr>
<tr>
<td>BarActiveLabels</td>
<td>BarActiveLabels = (&quot;Label&quot;, &quot;URL&quot;, &quot;Target&quot;), ...;</td>
</tr>
<tr>
<td>BarBorder</td>
<td>BarBorder = (LineType, LineWidth, Color);</td>
</tr>
<tr>
<td>BarColorTable[n]</td>
<td>BarColorTable[1-50] = color1,color2...;</td>
</tr>
<tr>
<td>BarColorTable[n][m]</td>
<td>BarColorTable[1-50][1-50] = color1,color2...;</td>
</tr>
<tr>
<td>BarCorners</td>
<td>BarCorners = (topleft, topright, bottomright, bottomleft);</td>
</tr>
</tbody>
</table>
BarDropShadow

BarFillPattern

BarFillPattern[n][m]

BarHighlights

BarRightFillPattern

BarRightFillPattern[n][m]

BarSpotlights

BarSymbol

BarTopFillPattern

BarTopFillPattern[n][m]

BarValueLabel

BarValueLabelBox

BarValueLabelStyle

BarWidth

DataAxis

DataSets

DataSet[n]

DataSet[n][m]

DataLegend

DataLegendGrid

DataLegendGridLine

GraphType

GraphLayout

GroupStackLabels

GroupStackSegmentLabels

PlotArea

ShowGroupStackLabels

StackDisplayOrder

StackedBarConnectors

StackLabel

ViewPoint

ZAxisLabels
## Box Chart

### Generally Supported Parameter Types

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Axis</td>
<td>See Chapter 6, <em>Axis Modifications</em> for various parameters available</td>
</tr>
<tr>
<td>Axis Thickness</td>
<td>AxisThickness = 15;</td>
</tr>
<tr>
<td>Color Table</td>
<td>See Chapter 8, <em>Color</em> for application of color attributes</td>
</tr>
<tr>
<td>BuildAnimation Enabled</td>
<td>BuildAnimationEnabled = ON</td>
</tr>
<tr>
<td>Background</td>
<td>Background = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor);</td>
</tr>
<tr>
<td>BackgroundFillPattern</td>
<td>BackgroundFillPattern = (type, color1, color2, imageURL), ...;</td>
</tr>
<tr>
<td>ChartElementSpacing</td>
<td>ChartElementSpacing = spacing;</td>
</tr>
<tr>
<td>Grid</td>
<td>Grid = (LineColor1, bgColor1, borderColor1, bgImage1, ImageFormat1), ...;</td>
</tr>
<tr>
<td>Legend</td>
<td>Legend = (&quot;Label&quot;, Color, &quot;FontName&quot;, FontSize, Angle, interiorAlignment);</td>
</tr>
<tr>
<td>Note Sets</td>
<td>NoteSets = (&quot;Name1&quot;, Justify1), (&quot;Name2&quot;, Justify2), ...;</td>
</tr>
<tr>
<td>Title</td>
<td>See Chapter 6, <em>Labels</em> for various parameters available.</td>
</tr>
<tr>
<td>TitleBox</td>
<td>See Chapter 6, <em>Regions, or Boxes</em> for various parameters available.</td>
</tr>
</tbody>
</table>

### Specifically Supported Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BoxActiveLabels</td>
<td>BoxActiveLabels = (&quot;Label1&quot;,&quot;URL1&quot;,&quot;Target1&quot;), ...;</td>
</tr>
<tr>
<td>BoxFence</td>
<td>BoxFence = ON</td>
</tr>
<tr>
<td>BoxFillPattern</td>
<td>BoxFillPattern = (type, color1, color2, imageURL), ...;</td>
</tr>
<tr>
<td>BoxHeight</td>
<td>BoxHeight = Height;</td>
</tr>
<tr>
<td>BoxLabels</td>
<td>BoxLabels = &quot;label1&quot;,&quot;label2&quot;,...</td>
</tr>
<tr>
<td>BoxLimitLines</td>
<td>BoxLimitLines = (limit1-1,limit1-2,...limit1-N),... (limitM-1, limitM-2,... limitM-N);</td>
</tr>
<tr>
<td>BoxLimitLineStyle</td>
<td>BoxLimitLineStyle = (type1,width1,color1),... (typeN,widthN,colorN);</td>
</tr>
<tr>
<td>BoxSymbolWidth</td>
<td>BoxSymbolWidth = Percent;</td>
</tr>
<tr>
<td>BarWidth</td>
<td>BarWidth = Percent1,Percent2,...</td>
</tr>
<tr>
<td>DataAxis</td>
<td>DataAxis = (XAxis1, YAxis1), (XAxis2, YAxis2), ...;</td>
</tr>
<tr>
<td>DataPointActiveLabels</td>
<td>DataPointActiveLabels(n) = (&quot;Label1&quot;,&quot;URL1&quot;,&quot;Target1&quot;), ...;</td>
</tr>
<tr>
<td>DataPointColor</td>
<td>DataPointColor = Color;</td>
</tr>
<tr>
<td>DataPointJitter</td>
<td>DataPointJitter = ON</td>
</tr>
<tr>
<td>DataPointSymbol</td>
<td>DataPointSymbol = (type1.size1.style1,.borderColor1, borderwidth1,image1,color1), ...;</td>
</tr>
<tr>
<td>DataType</td>
<td>DataType = Type;</td>
</tr>
<tr>
<td>DataSets</td>
<td>DataSets = (DataSetName1, Color1, BorderType1, BorderWidth1, &quot;ImageURL&quot;, ImageFormat1, BorderColor1), ...;</td>
</tr>
</tbody>
</table>
DataSet[n] = a, b, ...;
FenceActiveLabels[1-50] = ("Label1", "URL1", "Target1"), ...;
FencePosition = OVER | UNDER;
GraphLayout = VERTICAL | HORIZONTAL;
MeanActiveLabels = ("Label1", "URL1", "Target1"), ...;
MeanColor
MeanLine
MeanSymbol = (type1, size1, style1, bordercolor1, borderwidth1, image1, color1), ...;
MedianColor
NaturalDisplayOrder
MinimumDataPoints = int_val;
OutlierActiveLabels[1-50] = ("Label1", "URL1", "Target1"), ...;
OutlierColor
OutlierSymbol = (type1, size1, style1, bordercolor1, borderwidth1, image1, color1), ...;
PercentileN
PlotArea
PlotType
RelativeBoxSymbolWidth
ShowDataPoints
WhiskerType

**Bubble Chart**

**Generally Supported Parameter Types**

ActiveLabels[1-50] = ("Label1", "URL1", "Target1"), ...

See also Chapter 6, *Active Labels and Drill-Down*

Axis
Color Table
Background
BackgroundFillPattern
BubbleAnimationStyle
BuildAnimationEnabled
ChartElementSpacing
Grid
Legend

CDL Reference Manual – 7.1

© Visual Mining, Inc 2002-2011
www.visualmining.com
### CDL Parameter Arranged by Chart Style

**Note Sets**

`NoteSets = ("Name1", Justify1), ("Name2", Justify2), ...;`

See also Chapter 6, Notes, or Annotations

**Title**

See Chapter 6, Labels for various parameters available.

**TitleBox**

See Chapter 6, Regions, or Boxes for various parameters available.

### Specifically Supported Parameters

- **AddDataPoint**
  
  `AddDataPoint = ("Name", X, Y, Z, "Label", "URL", "Target"), ...;`

- **BubbleAxis**
  
  `BubbleAxis = (XAxis1, YAxis1), (XAxis2, YAxis2), ...;`

- **BubbleColorTable[n]**
  
  `BubbleColorTable[1-50] = color1, color2...;`

- **BubbleFillPattern**
  
  `BubbleFillPattern = (type, color1, color2, imageURL), ...;`

- **BubbleScale**
  
  `BubbleScale = (MinValue, MaxValue, AREA|DIAMETER, PointColor), ...;`

- **BubbleSets[n]**
  
  `BubbleSetsn = (x,y,z), (x,y,z), ...;`

- **BubbleSymbol**
  
  `BubbleSymbol = (SymType, MaxSize, Style, BorderColor, BorderWidth, SymbolColor, ShadowWidth), ...;`

- **LineStyle**
  
  `LineStyle = (LineType, LineWidth, Color, FillType, LineType, FillType), ...;`

- **PlotArea**
  
  `PlotArea = (xlocation, ylocation, width, height);`

### Combo Chart

**Generally Supported Parameter Types**

- **ActiveLabels[1-50]**
  
  `ActiveLabels[1-50] = ("Label1", "URL1", "Target1"), ...;`

  See also Chapter 6, Active Labels and Drill-Down

- **Axis**
  
  `Axis Thickness = 15;`

- **BuildAnimationEnabled**
  
  `BuildAnimationEnabled = ON|OFF;`

- **Color Table**
  
  `See Chapter 8, Color for application of color attributes`

- **Background**
  
  `Background = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor);`

- **BackgroundFillPattern**
  
  `BackgroundFillPattern = (type, color1, color2, imageURL), ...;`

- **ChartElementSpacing**
  
  `ChartElementSpacing = spacing;`

- **DataLegend**
  
  `DataLegend = ON|OFF;`

- **DataLegendGrid**
  
  `DataLegendGrid = (lineColor, bgColor, borderColor, bgImage, bgImageType);`

- **DataLegendGridLine**
  
  `DataLegendGridLine = (lineType1, lineStyle1, lineWidth1);`

- **Grid**
  
  `Grid = (LineColor1, bgColor1, borderColor1, bgImage1, ImageFormat1), ...;`

- **Legend**
  
  `Legend = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment);`

- **Note Sets**
  
  `NoteSets = ("Name1", Justify1), ("Name2", Justify2), ...;`

  See also Chapter 6, Notes, or Annotations
Specifically Supported Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BarFillPattern</td>
<td>BarFillPattern = (type, color1, color2, imageURL), ...;</td>
</tr>
<tr>
<td>Bar3DDepth</td>
<td>Bar3DDepth = Number;</td>
</tr>
<tr>
<td>BarSymbol</td>
<td>BarSymbol = (BarSymbolType, BarColor);</td>
</tr>
<tr>
<td>BarValueLabel</td>
<td>BarValueLabel = (mode, color, font name, width);</td>
</tr>
<tr>
<td>BarValueLabelBox</td>
<td>BarValueLabelBox = (color, mode, depth);</td>
</tr>
<tr>
<td>BarValueLabelStyle</td>
<td>BarValueLabelStyle = labelposition1, labelposition2, ...;</td>
</tr>
<tr>
<td>BarActiveLabels</td>
<td>BarActiveLabels = (&quot;Label&quot;, &quot;URL&quot;, &quot;Target&quot;), ...;</td>
</tr>
<tr>
<td>BarBorder</td>
<td>BarBorder = (LineType, LineWidth, Color);</td>
</tr>
<tr>
<td>BarColorTable[n]</td>
<td>BarColorTable[1-50] = color1, color2...;</td>
</tr>
<tr>
<td>BarWidth</td>
<td>BarWidth = Percent1, Percent2,...;</td>
</tr>
<tr>
<td>DataAxis</td>
<td>DataAxis = (XAxis1, YAxis1), (XAxis2, YAxis2), ...;</td>
</tr>
<tr>
<td>DataSets</td>
<td>DataSets = (DataSetName1, Color1, BorderType1, BorderWidth1, &quot;ImageURL&quot;, ImageFormat1, BorderColor1), ...;</td>
</tr>
<tr>
<td>DataSet[n]</td>
<td>DataSet[1-50] = a, b, c, ...;</td>
</tr>
<tr>
<td>DrawOrder</td>
<td>DrawOrder = Symbol;</td>
</tr>
<tr>
<td>GraphType</td>
<td>GraphType = Type;</td>
</tr>
<tr>
<td>GraphLayout</td>
<td>GraphLayout = Type;</td>
</tr>
<tr>
<td>PlotArea</td>
<td>PlotArea = (xlocation, ylocation, width, height);</td>
</tr>
<tr>
<td>StackLabel</td>
<td>StackLabel = Type;</td>
</tr>
<tr>
<td>Line3DDepth</td>
<td>Line3Ddepth = depth;</td>
</tr>
<tr>
<td>LineAxis</td>
<td>LineAxis = (XAxis1, YAxis1), (XAxis2, YAxis2), ...;</td>
</tr>
<tr>
<td>LineColorTable[n]</td>
<td>LineColorTable[1-50] = color1, color2...;</td>
</tr>
<tr>
<td>LineDropShadow</td>
<td>LineDropShadow = (color, offsetx, offsety, size);</td>
</tr>
<tr>
<td>LineFillPattern</td>
<td>LineFillPattern = (type, color1, color2, imageURL), ...;</td>
</tr>
<tr>
<td>LineLabels[n]</td>
<td>LineLabels[1-50] = (&quot;Label&quot;, &quot;URL&quot;, &quot;Target&quot;), ...;</td>
</tr>
<tr>
<td>LineSets</td>
<td>LineSets = (Name1, SymColor1), (Name2, SymColor2), ...;</td>
</tr>
<tr>
<td>LineSet[n]</td>
<td>LineSet[1-50] = y1, y2, y3,...;</td>
</tr>
<tr>
<td>LineStyle</td>
<td>LineStyle = (LineType, LineWidth, Color, FillColor, LineType, FillType), ...;</td>
</tr>
<tr>
<td>LineSymbol</td>
<td>LineSymbol = (Type, Size, Style, BorderColor, BorderWidth, ImageURL, SymbolColor, ShadowWidth), ...;</td>
</tr>
<tr>
<td>Spotlights</td>
<td>Spotlights = (start, stop, center, centeroffsetx, centeroffsety, focusoffsetx, focusoffsety, radius), ...;</td>
</tr>
<tr>
<td>LineValueLabel</td>
<td>LineValueLabel = (mode, color, font name, width);</td>
</tr>
<tr>
<td>LineValueLabelBox</td>
<td>LineValueLabelBox = (color, mode, depth);</td>
</tr>
<tr>
<td>LineValueLabelStyle</td>
<td>LineValueLabelStyle = labelposition1, labelposition2, ...;</td>
</tr>
<tr>
<td>LineWidth</td>
<td>LineWidth = PercentDepth;</td>
</tr>
</tbody>
</table>
StackDisplayOrder
StackDisplayOrder = mode;

StackedBar
StackedBarConnectors = OFF | LINE | FILL;

Connectors
StackLabel
StackLabel = Type;

**Diagram/Map Chart**

**Supported Parameter Types**

<table>
<thead>
<tr>
<th>Parameter Type</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveLabels</td>
<td>ActiveLabels = (&quot;Label1&quot;, &quot;URL1&quot;, &quot;Target1&quot;), ...; See also Chapter 6, Active Labels and Drill-Down</td>
</tr>
<tr>
<td>Color Table</td>
<td>See Chapter 8, Color for application of color attributes</td>
</tr>
<tr>
<td>Background</td>
<td>Background = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor)</td>
</tr>
<tr>
<td>BackgroundFillPattern</td>
<td>BackgroundFillPattern = (type, color1, color2, imageURL), ...;</td>
</tr>
<tr>
<td>Legend</td>
<td>Legend = (&quot;Label&quot;, Color, &quot;FontName&quot;, FontSize, Angle, interiorAlignment);</td>
</tr>
<tr>
<td>Note Sets</td>
<td>NoteSets = (&quot;Name1&quot;, Justify1), (&quot;Name2&quot;, Justify2), ...;</td>
</tr>
<tr>
<td>Title</td>
<td>See Chapter 6, Notes, or Annotations</td>
</tr>
<tr>
<td>TitleBox</td>
<td>See Chapter 6, Regions, or Boxes for various parameters available.</td>
</tr>
</tbody>
</table>

**Specifically Supported Parameters**

<table>
<thead>
<tr>
<th>Parameter Type</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>AppendPolyDataToActiveLabels</td>
<td>AppendPolyDataToActiveLabels = ON</td>
</tr>
<tr>
<td>Edges</td>
<td>Edges = (NodeStart, NodeEnd, Color, Direction, LineStyle, LineWidth, ArrowStyle, ArrowLength, ArrowWidth), ...;</td>
</tr>
<tr>
<td>NodeBox</td>
<td>NodeBox = (Color, BorderType, BorderWidth, ImageURL, ImageFormat, BorderColor)</td>
</tr>
<tr>
<td>NodeDrag</td>
<td>NodeDrag = ON</td>
</tr>
<tr>
<td>NodeLabel</td>
<td>NodeLabel = (Color, FontName, FontSize, Angle), ...;</td>
</tr>
<tr>
<td>Nodes</td>
<td>Nodes = (Name, Label, X, Y), ...;</td>
</tr>
<tr>
<td>PolyActiveLabels</td>
<td>PolyActiveLabels = (&quot;Label1&quot;, &quot;URL1&quot;, &quot;Target1&quot;), ...; See also Chapter 6, Active Labels and Drill-Down</td>
</tr>
<tr>
<td>PolyColor</td>
<td>PolyColor = (tagName, color), ...;</td>
</tr>
<tr>
<td>PolySet</td>
<td>PolySet = (tagName, x1,y1,x2,y2,...), ...;</td>
</tr>
</tbody>
</table>

**Dial Chart**

**Generally Supported Parameter Types**

<table>
<thead>
<tr>
<th>Parameter Type</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color Table</td>
<td>See Chapter 8, Color for application of color attributes</td>
</tr>
<tr>
<td>Background</td>
<td>Background = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;, BorderColor);</td>
</tr>
</tbody>
</table>
BackgroundFillPattern  
BackgroundFillPattern = (type, color1, color2, imageURL), ...;

BuildAnimation  
BuildAnimationEnabled = ON|OFF;

ChartElementSpacing  
ChartElementSpacing = spacing;

Legend  
Legend = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment);

Note Sets  
NoteSets = ("Name1", Justify1), ("Name2", Justify2), ...;

Title  
See also Chapter 6, Notes, or Annotations

TitleBox  
See Chapter 6, Regions, or Boxes for various parameters available.

Specifically Supported Parameters

Dials  
Dials = (Name, StartAngle, StopAngle, RadiusPercentage, NONE\inside\outside), ...;

DialActiveLabels  
DialActiveLabels = (Name, Label, URL, Target), ...;

DialBorders  
DialBorders = (Name, Type, Thickness, Color, NONE\center\endtoend), ...;

DialClip  
DialClip = clipType;

DialClipPad  
DialClipPad = N;

DialDelete  
DialDelete = (Name\all), ...;

DialFills  
DialFills = (Name, Color, NONE\center\endtoend), ...;

DialFillPattern  
DialFillPattern = (type, color1, color2, imageURL), ...;

DialFormats  
DialFormats = (Name, FLOAT\integer\decimal, formatExpression), ...;

DialHandAnimationStyle  
DialHandAnimationStyle = GROW | FADE | NONE

DialOuterBorder  
DialOuterBorder = (color1, color2, width);

DialOuterFillPattern  
DialOuterFillPattern = (type, color1, color2, imageURL);

DialScale  
DialScale = (Name, MinValue, MaxValue, StepValue), ...;

DialSize  
DialSize = (minWidth, minHeight, maxWidth, maxHeight);

DialSectorAnimationStyle  
DialSectorAnimationStyle = GROW | FADE | NONE

DialSquare  
DialSquare = Switch;

DialTics  
DialTics = (Name, Color, LineWidth, PercentofRadius), ...;

DialTicLabels  
DialTicLabels = (Name, Label1, Label2, ..., LabelN), ...;

DialTicLabelStyles  
DialTicLabelStyles = (Name, ON\OFF, LabelPos, Color, FontName, FontSize, Angle, interiorAlignment), ...;

Hands  
Hands = (Name, TipColor, ShaftColor, DialName, HandLabel), ...;

HandActiveLabels  
HandActiveLabels = (Name, Label, URL, Target), ...;

HandBorders  
HandBorders = (Name, lineType, LineWidth, Color);

HandButtonBorder  
HandButtonBorder = (lineType, LineWidth, Color);

HandButton  
HandButtonEdgeHighlights = (start, stop, gap, size), ...;

HandData  
HandData = (Name, Value, Length), ...;
HandDelete

HandDelete = (Name|ALL), ...;

HandDrag

HandDrag = "ON"|"OFF";

HandDropShadow

HandDropShadow = (color, offsetx, offset, size);

HandLabels

HandLabels = ("Name", "ON|OFF", labelPos, color, font, size, angle, justification), ...;

HandStyles

HandStyles = (Name, NEEDLELINE|NEEDLEFILL|NEEDLEBUTTON|SHARP|ROUND|BLOCK|LINE|NONE, TipWidth, ShaftWidth), ...;

Sectors

Sectors = (Name, Color, DialName, OuterRadius, InnerRadius, SectorLabel), ...;

SectorActiveLabels

SectorActiveLabels = (Name, Label, URL, Target), ...;

SectorBorders

SectorBorders = (Name, Type, LineWidth, Color), ...;

SectorColors

SectorColors = (Name, color), ...;

SectorData

SectorData = (Name, StartValue, StopValue), ...;

SectorDelete

SectorDelete = (Name|ALL), ...;

SectorDrag

SectorDrag = "ON"|"OFF";

SectorEdgeHighlights

SectorEdgeHighlights = (start, stop, gap, size), ...;

SectorFillPattern

SectorFillPattern = (type, color1, color2, imageURL), ...;

SectorLabels

SectorLabels = (Name, ON|OFF, LabelPos, Color, FontName, FontSize, Angle, interiorAlignment), ...;

Heat Map

Generally Supported Parameter Types

ChartElementSpacing

ChartElementSpacing = spacing;

Legend

Legend = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment);

Note Sets

NoteSets = ("Name1", Justify1), ("Name2", Justify2), ...;

See also Chapter 6, Notes, or Annotations

Title

See Appendix A, Labels for various parameters available.

TitleBox

See Chapter 6, Regions, or Boxes for various parameters available.

Specifically Supported Parameters

CellTextAutoColorThreshold

CellTextAutoColorThreshold = 0-100 %;

GridBlockActiveLabels

GridBlockActiveLabels = (Name, Label, URL, Target), ...;

GridBlockBackgroundColor

GridBlockBackgroundColor = color;

GridBlockCellColorType

GridBlockCellColorType = type;

GridBlockColors

GridBlockColors = (color1, color2,...,colorN);

GridBlockColors

GridBlockColors = (color1, color2,...,colorN);
Generally Supported Parameter Types

ActiveLabels[1-50] = ("Label1", "URL1", "Target1")
See also Chapter 6, Active Labels and Drill-Down

Axis Thickness = 15;
Color Table
See Chapter 8, Color for application of color attributes
Background = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor);
BackgroundFillPattern = (type, color1, color2, imageURL), ...
BuildAnimationEnabled = ON|OFF;
ChartElementSpacing = spacing;
Grid = (LineColor1, bgColor1, borderColor1, bgImage1, ImageFormat1), ...
Legend = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment);
Note Sets = ("Name1", Justify1), ("Name2", Justify2), ...
See also Chapter 6, Notes, or Annotations
Title
See Chapter 6,
Labels for various parameters available.
TitleBox
See Chapter 6, Regions, or Boxes for various parameters available.

### Specifically Supported Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bar3DDepth</td>
<td>Bar3DDepth = Number;</td>
</tr>
<tr>
<td>BarActiveLabels</td>
<td>BarActiveLabels = (&quot;Label&quot;, &quot;URL&quot;, &quot;Target&quot;), ...;</td>
</tr>
<tr>
<td>BarBorder</td>
<td>BarBorder = (LineType, LineWidth, Color);</td>
</tr>
<tr>
<td>BarColorTable[n]</td>
<td>BarColorTable[1-50] = color1,color2...;</td>
</tr>
<tr>
<td>BarColorTable[n]P[m]</td>
<td>BarColorTable[1-50]P[1-50] = color1,color2...;</td>
</tr>
<tr>
<td>BarFillPattern</td>
<td>BarFillPattern = (type, Color1, Color2, imageURL), ...;</td>
</tr>
<tr>
<td>BarFillPattern [1-50]P[1-50]</td>
<td>(for STACKEDGROUPED BAR)</td>
</tr>
<tr>
<td>BarValueLabel</td>
<td>BarValueLabel = (mode, color, font name, width);</td>
</tr>
<tr>
<td>BarValueLabelBox</td>
<td>BarValueLabelBox = (color, mode, depth);</td>
</tr>
<tr>
<td>BarValueLabelStyle</td>
<td>BarValueLabelStyle = labelposition1,labelposition2,...;</td>
</tr>
<tr>
<td>BarWidth</td>
<td>BarWidth = Percent1,Percent2,...;</td>
</tr>
<tr>
<td>DataAxis</td>
<td>DataAxis = (XAxis1, YAxis1), (XAxis2, YAxis2), ...;</td>
</tr>
<tr>
<td>DataSets</td>
<td>DataSets = (Label1, Color1, Type1), (Label2, Color2, Type2), ...;</td>
</tr>
<tr>
<td>DataSet[n]</td>
<td>DataSet[1-50] = a, b, c, ...;</td>
</tr>
<tr>
<td>DataSet[n]P[m]</td>
<td>DataSet[1-50]P[1-50] = a, b, c, ...;</td>
</tr>
<tr>
<td>GraphType</td>
<td>GraphType = Type;</td>
</tr>
<tr>
<td>GraphLayout</td>
<td>GraphLayout = Type;</td>
</tr>
<tr>
<td>GroupStackLabels</td>
<td>GroupStackLabels = &quot;label1&quot;,&quot;label2&quot;,...;</td>
</tr>
<tr>
<td>GroupStackSegmentLabels</td>
<td>GroupStackSegmentLabels = &quot;label1&quot;,&quot;label2&quot;,...;</td>
</tr>
<tr>
<td>HistogramBin</td>
<td>HistogramBin = (HistogramBinType, HistogramBinSize);</td>
</tr>
<tr>
<td>HistogramScale</td>
<td>HistogramScale = (histogramMinValue, histogramMaxValue);</td>
</tr>
<tr>
<td>HistogramType</td>
<td>HistogramType = BYNUMBER</td>
</tr>
<tr>
<td>PlotArea</td>
<td>PlotArea = (slocation, ylocation, width, height);</td>
</tr>
<tr>
<td>ShowGroupStackLabels</td>
<td>ShowGroupStackLabels = ON</td>
</tr>
<tr>
<td>StackDisplayOrder</td>
<td>StackDisplayOrder = mode;</td>
</tr>
<tr>
<td>StackLabel</td>
<td>StackLabel = Type;</td>
</tr>
</tbody>
</table>

### Line Chart

#### Generally Supported Parameter Types

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Axis</td>
<td>See also Chapter 6, Active Labels and Drill-Down</td>
</tr>
</tbody>
</table>

See also Chapter 6, Axis Modifications for various parameters available.
Axis Thickness

AxisThickness = 15;

Build Animation

BuildAnimationEnabled = ON|OFF;

Color Table

See Chapter 8, Color for application of color attributes

Background

Background = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor);

Background Fill Pattern

BackgroundFillPattern = (type, color1, color2, imageURL), ...;

Build Animation

BuildAnimationEnabled = ON|OFF;

Chart Element Spacing

ChartElementSpacing = spacing;

Grid

Grid = (LineColor1, bgColor1, borderColor1, bgImage1, ImageFormat1), ...

Legend

Legend = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment);

Note Sets

NoteSets = ("Name1", Justify1), ("Name2", Justify2), ...

See also Chapter 6, Notes, or Annotations

Title

See Chapter 6, Labels for various parameters available.

Title Box

See Chapter 6, Regions, or Boxes for various parameters available.

Specifically Supported Parameters

Chart Element Spacing

ChartElementSpacing = spacing;

Data Legend

DataLegend = ON|OFF;

Data Legend Grid

DataLegendGrid = (lineColor, bgColor, borderColor, bgImage, ImageType);

Data Legend Grid Line

DataLegendGridLine = (lineType1, lineStyle1, lineWidth1);

Dwell Offset

DwellOffset = size;

Graph Type

GraphType = Type;

Line 3D Depth

Line3DDepth = depth;

Line Axis

LineAxis = (XAxis1, YAxis1), (XAxis2, YAxis2), ...

Line Fill Pattern

LineFillPattern = (type, color1, color2, imageURL), ...

Line Sets

LineSets = (Name1, SymColor1), (Name2, SymColor2), ...

Line Set [1-50]

LineSet[1-50] = y1, y2, y3, ...

Line Style

LineStyle = (LineType, Line Width, Color, Fill Color, Line Type, Fill Type), ...

Line Symbol

LineSymbol = (Type, Size, Style, BorderColor, Border Width, ImageURL, Symbol Color, Shadow Width), ...

Line Symbol Spotlights

LineSymbolSpotlights = (start, stop, center, centeroffsetx, centeroffsety, focusoffsetx, focusoffsety, radius), ...

Line Value Label Box

LineValueLabelBox = (color, mode, depth);

Line Value Label Style

LineValueLabelStyle = labelposition1, labelposition2, ...

LineWidth

LineWidth = PercentDepth;

Plot Area

PlotArea = (xlocation, ylocation, width, height);

Stack Label

StackLabel = Type;
**MultiPie Chart**

**Generally Supported Parameter Types**

- **ActiveLabels**
  
  ```
  ActiveLabels = ("Label1", "URL1", "Target1"), ...
  ```
  
  See also Chapter 6, Active Labels and Drill-Down

- **Axis**
  
  See Chapter 6, Axis Modifications for various parameters available

- **BuildAnimation**
  
  ```
  BuildAnimationEnabled = ON|OFF;
  ```

- **Color Table**
  
  See Chapter 8, Color for application of color attributes

- **ChartElementSpacing**
  
  ```
  ChartElementSpacing = spacing;
  ```

- **Legend**
  
  ```
  Legend = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment);
  ```

- **Note Sets**
  
  ```
  NoteSets = ("Name1", Justify1), ("Name2", Justify2), ...
  ```
  
  See also Chapter 6, Notes, or Annotations

- **Title**
  
  See Chapter 6, Labels for various parameters available.

- **TitleBox**
  
  See Chapter 6, Regions, or Boxes for various parameters available.

**Specifically Supported Parameters**

- **BestFit**
  
  ```
  BestFit = ON|OFF
  ```

- **LabelPos**
  
  ```
  LabelPos = Float;
  ```

- **Pie3DDepth**
  
  ```
  Pie3DDepth = Pixels;
  ```

- **PieAngles**
  
  ```
  PieAngles = Value1, Value2, Value3, Value4, ..., ValueN;
  ```

- **PieDropShadow**
  
  ```
  PieDropShadow = (color, offsetX, offsetY, size);
  ```

- **PieEdgeHighlights**
  
  ```
  PieEdgeHighlights = (start, stop, gap, size), ...
  ```

- **PieSquare**
  
  ```
  PieSquare = ON/OFF;
  ```

- **PieBackgrounds**
  
  ```
  PieBackgrounds = (Region Tuple), ...
  ```

- **PieLabel**
  
  ```
  PieLabel = (State, Color, FontName, FontSize, Angle, InteriorAlignment), ...
  ```

- **PieLabelBox**
  
  ```
  PieLabelBox = (Region Tuple), ...
  ```

- **PieLayout**
  
  ```
  PieLayout = (Orientation, Row, Columns);
  ```

- **PieLabelLocation**
  
  ```
  PieLabelLocation = Location;
  ```

- **PieMargin**
  
  ```
  PieMargin = Integer;
  ```

- **PieLabels**
  
  ```
  PieLabels = Label1, Label2, Label3, ... LabelN;
  ```

- **SliceBorder**
  
  ```
  SliceBorder = (LineType, Width, Color);
  ```

- **SliceSets**
  
  ```
  SliceSets = ("Name", Color, "State");
  ```

- **SliceSet[N]**
  
  ```
  SliceSet = Value1, Value2, Value3, Value4, ..., ValueN;
  ```

- **SliceFillPattern**
  
  ```
  SliceFillPattern = (type, color1, color2, imageURL), ...
  ```

- **SliceFormat**
  
  ```
  SliceFormat = (FormatType, "FormatExpr")
  ```
SliceLabel

\[
\text{SliceLabel} = (\text{State, Color, FontName, FontSize, Angle, interiorAlignment});
\]

SliceLabelBox

\[
\text{SliceLabelBox} = (\text{Color, BorderType, BorderWidth});
\]

SliceLabelContent

\[
\text{SliceLabelContent} = \text{Content};
\]

SliceLabelContentDelimiter

\[
\text{SliceLabelContentDelimiter} = \text{"delimiter"};
\]

SliceLabelLine

\[
\text{SliceLabelLine} = (\text{LineStyle, LineWidth, Color});
\]

SliceLabels

\[
\text{SliceLabels} = \text{Label1, Label2, Label3, Label4, \ldots, LabelN};
\]

SliceLabelStyle

\[
\text{SliceLabelStyle} = (\text{Style});
\]

SlicePos[N]

\[
\text{SlicePos}[N] = \text{PiePosition1, PiePosition2, \ldots};
\]

### Pareto Chart

**Generally Supported Parameter Types**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveLabels[1-50]</td>
<td>[&quot;Label1&quot;, &quot;URL1&quot;, &quot;Target1&quot;], \ldots; See also Chapter 6, Active Labels and Drill-Down</td>
</tr>
<tr>
<td>Axis Thickness</td>
<td>\text{AxisThickness} = 15;</td>
</tr>
<tr>
<td>Color Table</td>
<td>See Chapter 8, Color for application of color attributes</td>
</tr>
<tr>
<td>BuildAnimation Enabled</td>
<td>\text{BuildAnimationEnabled} = \text{ON</td>
</tr>
<tr>
<td>Background</td>
<td>\text{Background} = (\text{Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor});</td>
</tr>
<tr>
<td>BackgroundFillPattern</td>
<td>\text{BackgroundFillPattern} = (type, color1, color2, imageURL), \ldots;</td>
</tr>
<tr>
<td>ChartElementSpacing</td>
<td>\text{ChartElementSpacing} = \text{spacing};</td>
</tr>
<tr>
<td>Grid</td>
<td>\text{Grid} = (\text{LineColor1, bgColor1, borderColor1, bgImage1, ImageFormat1}), \ldots;</td>
</tr>
<tr>
<td>Legend</td>
<td>\text{Legend} = (&quot;Label&quot;, Color, &quot;FontName&quot;, FontSize, Angle, interiorAlignment);</td>
</tr>
<tr>
<td>Note Sets</td>
<td>\text{NoteSets} = (&quot;Name1&quot;, Justify1), (&quot;Name2&quot;, Justify2), \ldots;</td>
</tr>
<tr>
<td>Title</td>
<td>See Chapter 6, Notes, or Annotations</td>
</tr>
<tr>
<td>TitleBox</td>
<td>See Chapter 6, Regions, or Boxes for various parameters available.</td>
</tr>
</tbody>
</table>

**Specifically Supported Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BarFillPattern</td>
<td>\text{BarFillPattern} = (type, color1, color2, imageURL), \ldots;</td>
</tr>
<tr>
<td>Bar3DDepth</td>
<td>\text{Bar3DDepth} = \text{Number};</td>
</tr>
<tr>
<td>BarValueLabel</td>
<td>\text{BarValueLabel} = (mode, color, font name, width);</td>
</tr>
<tr>
<td>BarValueLabelBox</td>
<td>\text{BarValueLabelBox} = (color, mode, depth);</td>
</tr>
<tr>
<td>BarValueLabelStyle</td>
<td>\text{BarValueLabelStyle} = labelposition1, labelposition2, \ldots;</td>
</tr>
<tr>
<td>BarActiveLabels</td>
<td>\text{BarActiveLabels} = (&quot;Label&quot;, &quot;URL&quot;, &quot;Target&quot;), \ldots;</td>
</tr>
<tr>
<td>BarBorder</td>
<td>\text{BarBorder} = (\text{LineType, LineWidth, Color});</td>
</tr>
</tbody>
</table>
BarColorTable[n]  BarColorTable[1-50] = color1, color2…;
BarWidth  BarWidth = Percent1, Percent2, …;
CumulativeLineSetName  CumulativeLineSetName = “name”;
CumulativeLineStyle  CumulativeLineStyle = (LineType, LineWidth, Color, FillColor);
CumulativeLineSymbol  CumulativeLineSymbol = (Type, Size, Style, BorderColor, BorderWidth, ImageURL, SymbolColor, ShadowWidth);
CumulativeLineValueLabel  CumulativeLineValueLabel = (mode, color, font name, width);
CumulativeLineValueLabelBox  CumulativeLineValueLabelBox = (color, mode, depth);
CumulativeLineValueLabelStyle  CumulativeLineValueLabelStyle = labelposition;
DataAxis  DataAxis = (XAxis1, YAxis1), (XAxis2, YAxis2), …;
DataSets  DataSets = (DataSetName1, Color1, BorderType1, BorderWidth1, "ImageURL", ImageFormat1, BorderColor1), …;
DataSet[n]  DataSet[1-50] = a, b, c, …;
DrawOrder  DrawOrder = Symbol;
EightyLineSetName  EightyLineSetName = “name”;
EightyTwentyLineStyle  EightyTwentyLineStyle = (LineType, LineWidth, Color, FillColor);
EightyTwentyLineSymbol  EightyTwentyLineSymbol = (Type, Size, Style, BorderColor, BorderWidth, ImageURL, SymbolColor, ShadowWidth);
Line3Ddepth  Line3Ddepth = depth;
LineSymbol  LineSymbolSpotlights = (start, stop, center, centeroffsetx, centeroffsety, focusoffsetx, focusoffsety, radius), …;
PlotArea  PlotArea = (xlocation, ylocation, width, height);
ShowEightTwentyLines  ShowEightTwentyLines = ON or OFF;
StackDisplayOrder  StackDisplayOrder = mode;
StackLabel  StackLabel = Type;
TwentyLineSetName  TwentyLineSetName = “name”;

**Pie Chart**

**Generally Supported Parameter Types**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveLabels</td>
<td>ActiveLabels = (“Label1”, “URL1”, “Target1”), …;</td>
</tr>
<tr>
<td>Axis</td>
<td>See Chapter 6, <em>Active Labels and Drill-Down</em></td>
</tr>
<tr>
<td>Color Table</td>
<td>See Chapter 8, <em>Color</em> for application of color attributes</td>
</tr>
<tr>
<td>Background</td>
<td>Background = (Color, BorderType, BorderWidth, “ImageURL”, ImageFormat, BorderColor);</td>
</tr>
<tr>
<td>BackgroundFillPattern</td>
<td>BackgroundFillPattern = (type, color1, color2, imageURL), …;</td>
</tr>
<tr>
<td>BuildAnimationEnabled</td>
<td>BuildAnimationEnabled = ON</td>
</tr>
<tr>
<td>ChartElementSpacing</td>
<td>ChartElementSpacing = spacing;</td>
</tr>
</tbody>
</table>

© Visual Mining, Inc 2002-2011
www.visualmining.com
Legend
Legend = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment);

Note Sets
NoteSets = ("Name1", Justify1), ("Name2", Justify2), ...
See also Chapter 6, Notes, or Annotations

Title
See Chapter 6,
Labels for various parameters available.

TitleBox
See Chapter 6, Regions, or Boxes for various parameters available.

**Specifically Supported Parameters**

**LabelPos**
LabelPos = Float;

**Pie3DDepth**
Pie3DDepth = Pixels;

**PieAngle**
PieAngle = Integer;

**PieDropShadow**
PieDropShadow = (color, offsetx, offsety, size);

**PieSize**
PieSize = (minWidth, minHeight, maxWidth, maxHeight);

**PieSquare**
PieSquare = Switch;

**SliceBorder**
SliceBorder = (LineType, Width, Color);

**SliceColor**
SliceColor = Color1, Color2, Color3 ..., Colorn;

**SliceData**
SliceData = Value1, Value2, Value3, Value4, ..., Valuen;

**SliceFillPattern**
SliceFillPattern = (type, color1, color2, imageURL), ...;

**SliceFormat**
SliceFormat = (FormatType, "FormatExpr");

**SliceLabel**
SliceLabel = (State, Color, FontName, FontSize, Angle, interiorAlignment);

**SliceLabelBox**
SliceLabelBox = (Color, BorderType, BorderWidth);

**SliceLabelContent**
SliceLabelContent = Content;

**SliceLabelContentDelimiter**
SliceLabelContentDelimiter = "delimiter";

**SliceLabelLine**
SliceLabelLine = (LineStyle, LineWidth, Color);

**SliceLabels**
SliceLabels = Label1, Label2, Label3, Label4, ..., Labeln;

**SliceLabelStyle**
SliceLabelStyle = (Style);

**SlicePos**
SlicePos = Position1, Position2, Positions3, ...;

**Slices**
Slices = (Value, SliceColor, Label, LabelColor, FontName, FontSize, LabelAngle, LabelBgColor, LabelBgBorder), ...;

**Polar Chart**

**Generally Supported Parameter Types**

**ActiveLabels[1-50]**
ActiveLabels[1-50] = ("Label", "URL1", "Target1"), ...
See also Chapter 6, Active Labels and Drill-Down

**Background**
Background = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor);

**BackgroundFillPattern**
BackgroundFillPattern = (type, color1, color2, imageURL), ...;

**ChartElementSpacing**
ChartElementSpacing = spacing;

**ColorTable**
See Chapter 8, Color for application of color attributes
Grid

Grid = (LineColor1, bgColor1, borderColor1, bgImage1, ImageFormat1), ...;

Legend

Legend = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment);

Note Sets

NoteSets = ("Name1", Justify1), ("Name2", Justify2), ...;

See also Chapter 6, Notes, or Annotations

Title

See Chapter 6, Labels for various parameters available.

TitleBox

See Chapter 6, Regions, or Boxes for various parameters available.

Specifically Supported Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LineSets</td>
<td>LineSets = (Name1, SymColor1), (Name2, SymColor2), ...;</td>
</tr>
<tr>
<td>LineSet[n]</td>
<td>LineSet[1-50] = (x1,y1), (x2,y2), (x3,y3), ...;</td>
</tr>
<tr>
<td>LineStyle</td>
<td>LineStyle = (LineType, LineWidth, Color, FillColor, LineType, FillType), ...;</td>
</tr>
<tr>
<td>LineSymbol</td>
<td>LineSymbol = (Type, Size, Style, BorderColor, BorderWidth, ImageURL, SymbolColor, ShadowWidth), ...;</td>
</tr>
<tr>
<td>LineValueLabel</td>
<td>LineValueLabel = (mode, color, font name, width);</td>
</tr>
<tr>
<td>LineValueLabelBox</td>
<td>LineValueLabelBox = (color, mode, depth);</td>
</tr>
<tr>
<td>LineValueLabelStyle</td>
<td>LineValueLabelStyle = labelposition1, labelposition2, ...;</td>
</tr>
<tr>
<td>PlotArea</td>
<td>PlotArea = (xlocation, ylocation, width, height);</td>
</tr>
<tr>
<td>PolarLabel</td>
<td>PolarLabel = (mode, color, font name, width);</td>
</tr>
<tr>
<td>PolarLabelFormat</td>
<td>PolarLabelFormat = (dataType, formatString);</td>
</tr>
<tr>
<td>PolarLabelStep</td>
<td>PolarLabelStep = stepsize;</td>
</tr>
<tr>
<td>PolarScale</td>
<td>PolarScale = (min, max, step);</td>
</tr>
<tr>
<td>PolarSquare</td>
<td>PolarSquare = ON</td>
</tr>
<tr>
<td>PolarSize</td>
<td>PolarSize = (minWidth, minHeight, maxWidth, maxHeight);</td>
</tr>
<tr>
<td>RadialAxesAngles</td>
<td>RadialAxesAngles = angle1,angle2,...;</td>
</tr>
<tr>
<td>RadialAxesColors</td>
<td>RadialAxesColors = color1,color2,...;</td>
</tr>
<tr>
<td>RadialAxesFormat</td>
<td>RadialAxesFormat = (dataType, formatString), ...;</td>
</tr>
<tr>
<td>RadialAxesTics</td>
<td>RadialAxesTics = (&quot;axisTicLabelMode&quot;,&quot;axisTicLabelColor&quot;,&quot;axisTicLabelFont&quot;,&quot;axisTicLabelFontSize&quot;,&quot;axisTicLabelFontAngle&quot;,interiorAlignment), ...;</td>
</tr>
<tr>
<td>RadialGrids</td>
<td>RadialGrids = (gridRadius,gridLineType,gridLineWidth,gridLineColor,gridAreaColor), ...;</td>
</tr>
</tbody>
</table>

### Radar Chart

**Generally Supported Parameter Types**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveLabels[1..50]</td>
<td>ActiveLabels[1..50] = (&quot;Label&quot;, &quot;URL1&quot;, &quot;Target1&quot;), ...;</td>
</tr>
<tr>
<td>Background</td>
<td>Background = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor);</td>
</tr>
<tr>
<td>BackgroundFillPattern</td>
<td>BackgroundFillPattern = (type, color1, color2, imageURL), ...;</td>
</tr>
</tbody>
</table>
CDL Parameter Arranged by Chart Style

ChartElementSpacing

ChartElementSpacing = spacing;

ColorTable

See Chapter 8, Color for application of color attributes

Grid

Grid = (LineColor1, bgColor1, borderColor1, bgImage1, ImageFormat1), ...;

Legend

Legend = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment);

Note Sets

NoteSets = ("Name1", Justify1), ("Name2", Justify2), ...;

See also Chapter 6, Notes, or Annotations

Title

See Chapter 6,

Labels for various parameters available.

TitleBox

See Chapter 6, Regions, or Boxes for various parameters available.

Specifically Supported Parameters

CenterRadius

CenterRadius = radius;

LineFillPattern

LineFillPattern = (type, color1, color2, imageUrl), ...;

LineSets

LineSets = (Name1, SymColor1), (Name2, SymColor2), ...;

LineSet[n]

LineSet[1-50] = y1, y2, y3, ...;

LineStyle

LineStyle = (LineType, LineWidth, Color, FillColor, LineType, FillType), ...;

LineSymbol

LineSymbol = (Type, Size, Style, BorderColor, BorderWidth, ImageURL, SymbolColor, ShadowWidth), ...;

LineValueLabel

LineValueLabel = (mode, color, font name, width);

LineValueLabelBox

LineValueLabelBox = (color, mode, depth);

LineValueLabelStyle

LineValueLabelStyle = labelposition1, labelposition2, ...;

PlotArea

PlotArea = (xlocation, ylocation, width, height);

RadarSize

RadarSize = (minWidth, minHeight, maxWidth, maxHeight);

RadarSquare

RadarSquare = mode;

RadialAxes

RadialAxes = ("axisTitle", minValue, maxValue, stepSize), ...;

RadialAxesColors

RadialAxesColors = color1, color2, ...;

RadialAxesFormat

RadialAxesFormat = (dataType, formatString), ...;

RadialAxesLabel

RadialAxesLabel = ("axisLabelMode", axisLabelColor, "axisLabelFont", axisLabelFontSize, axisLabelFontAngle, axisLabelInteriorAlignment), ...;

RadialAxesScales

RadialAxesScales = (min1, max1, step1), (min2, max2, step2), ...;

RadialAxesTics

RadialAxesTics = ("axisTicLabelMode", axisTicLabelColor, "axisTicLabelFont", axisTicLabelFontSize, axisTicLabelFontAngle, interiorAlignment), ...;

RadialAxesTitles

RadialAxesTitles = "title1", "title2", ...;

RadialGrids

RadialGrids = (gridRadius, gridLineType, gridLineWidth, gridLineColor, gridAreaColor), ...;
Stock Chart

Generally Supported Parameters

ActiveLabels[1-50] = ("Label1", "URL1", "Target1"), ...

See also Chapter 6, Active Labels and Drill-Down

Axis

Axis Thickness = 15;

BuildAnimationEnabled = ON|OFF;

Color Table

See Chapter 8, Color for application of color attributes

Background = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor);

BackgroundFillPattern = (type, color1, color2, imageURL), ...

ChartElementSpacing = spacing;

Grid = (LineColor1, bgColor1, borderColor1, bgColor1, ImageFormat1), ...

Legend = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment);

Note Sets = ("Name1", Justify1), ("Name2", Justify2), ...

See also Chapter 6, Notes, or Annotations

Title

See Chapter 6.

TitleBox

See Chapter 6, Regions, or Boxes for various parameters available.

Specifically Supported Parameters

Bar3DDepth = Number;

BarActiveLabels = ("Label", "URL", "Target"), ...

BarBorder = (LineType, LineWidth, Color);

BarFillPattern = (type, color1, color2, imageURL), ...

BarColorTable[1-50] = color1, color2...

BarWidth = Percent1,Percent2, ...

DataAxis = (XAxis1, YAxis1), (XAxis2, YAxis2), ...

DataSets = (DataSetName1, Color1, BorderType1, BorderWidth1, "ImageURL", ImageFormat1, BorderColor1), ...

DataSet[1-50] = a, b, c, ...

DrawOrder = Symbol, ...

GraphType = Type;

GraphLayout = Type;

GraphType = Type;

Line3DDepth = depth;

LineAxis = (XAxis1, YAxis1), (XAxis2, YAxis2), ...

LineColorTable[1-50] = color1, color2, ...

LineWidthTable = lineWidth1, lineWidth2, ...

LineFillPattern = (type, color1, color2, imageURL), ...
**Strip Chart**

### Generally Supported Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description and Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ActiveLabels [1-50]</strong></td>
<td><code>ActiveLabels[1-50] = (&quot;Label1&quot;, &quot;URL1&quot;, &quot;Target1&quot;)</code>; See also Chapter 6, Active Labels and Drill-Down</td>
</tr>
<tr>
<td><strong>Axis</strong></td>
<td>See Chapter 6, Axis Modifications for various parameters available</td>
</tr>
<tr>
<td><strong>Axis Thickness</strong></td>
<td><code>AxisThickness = 15;</code></td>
</tr>
<tr>
<td><strong>Color Table</strong></td>
<td>See Chapter 8, Color for application of color attributes</td>
</tr>
<tr>
<td><strong>Background</strong></td>
<td><code>Background = Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor</code></td>
</tr>
<tr>
<td><strong>BackgroundFillPattern</strong></td>
<td><code>BackgroundFillPattern = (type, color1, color2, imageURL), ...;</code></td>
</tr>
<tr>
<td><strong>ChartElementSpacing</strong></td>
<td><code>ChartElementSpacing = spacing;</code></td>
</tr>
<tr>
<td><strong>Grid</strong></td>
<td><code>Grid = (LineColor1, bgColor1, borderColor1, bgImage1, ImageFormat1), ...;</code></td>
</tr>
<tr>
<td><strong>Legend</strong></td>
<td><code>Legend = (&quot;Label&quot;, Color, &quot;FontName&quot;, FontSize, Angle, interiorAlignment);</code></td>
</tr>
</tbody>
</table>

**LineLabels[1-50]**

\[
\text{LineLabels}[1-50] = ("Label", "URL", "Target"), \ldots;
\]

**LineSets**

\[
\text{LineSets = (Name1, SymColor1), (Name2, SymColor2), \ldots;}
\]

**LineSet[n]**

\[
\text{LineSet}[1-50] = y1, y2, y3, \ldots;
\]

**LineStyle**

\[
\text{LineStyle = (LineType, LineWidth, Color, FillColor, LineType, FillType), \ldots;}
\]

**LineSymbol**

\[
\text{LineSymbol = (Type, Size, Style, BorderColor, BorderWidth, ImageURL, SymbolColor, ShadowWidth), \ldots;}
\]

**LineValueLabel**

\[
\text{LineValueLabel = (mode, color, font name, width);}\]

**LineValueLabelBox**

\[
\text{LineValueLabelBox = (color, mode, depth);}\]

**LineValueLabelStyle**

\[
\text{LineValueLabelStyle = labelposition1, labelposition2, \ldots;}\]

**LineWidth**

\[
\text{LineWidth = PercentDepth;}\]

**PlotArea**

\[
\text{PlotArea = (xlocation, ylocation, width, height);}\]

**StackDisplayOrder**

\[
\text{StackDisplayOrder = mode;}\]

**StackLabel**

\[
\text{StackLabel = Type;}\]

**StockAxis**

\[
\text{StockAxis = (XAxis1, YAxis1), (XAxis2, YAxis2), \ldots;}
\]

**StockData[1-50]**

\[
\text{StockData[1-50] = (High1, Low1, Open1, Close1), (High2, Low2, Open2, Close2), \ldots;}
\]

**StockColorTable[n]**

\[
\text{StockColorTable[1-50] = color1, color2...;}
\]

**StockFillPattern**

\[
\text{StockFillPattern = (type, color1, color2, imageURL), \ldots;}
\]

**StockLabels[1-50]**

\[
\text{StockLabels[1-50] = ("Label1", "URL1", "Target1"), ("Label2", "URL2", "Target2"), \ldots;}
\]

**StockSets**

\[
\text{StockSets = (Label1, Color1, Width1, TicLen1), (Label2, Color2, Width2, TicLen2), \ldots;}
\]

**StockWidth**

\[
\text{StockWidth = (Width, TicLen);}\]
Note Sets
NoteSets = ("Name1", Justify1), ("Name2", Justify2), ...
See also Chapter 6, Notes, or Annotations

Title
See Chapter 6,
Labels for various parameters available.

TitleBox
See Chapter 6, Regions, or Boxes for various parameters available.

**Specifically Supported Parameters**

- **AppendDataSet[n]**
  AppendDataSet[1-50] = y1, y2, y3, ...

- **BottomLabels**
  BottomLabels = "Label1", "Label2", ...

- **BottomScale**
  BottomScale = (MinValue, MaxValue, StepValue);

- **BottomScroll**
  BottomScroll = (ScrollMin, ScrollMax);

- **DataAxis**
  DataAxis = (XAxis1, YAxis1), (XAxis2, YAxis2), ...

- **DataSets**
  DataSets = (DataSetName1, Color1, BorderType1, BorderWidth1, "ImageURL", ImageFormat1, BorderColor1), ...
  DataSet[n] = a, b, c, ...

- **LineStyle**
  LineStyle = (LineType, LineWidth, Color, FillColor, LineType, FillType), ...

- **LineSymbol**
  LineSymbol = (Type, Size, Style, BorderColor, BorderWidth, ImageURL, SymbolColor, ShadowWidth), ...

- **LineSymbolSpotlights**
  LineSymbolSpotlights = (start, stop, center, centeroffsetx, centeroffsety, focusoffsetx, focusoffsety, radius), ...

- **LineValueLabelBox**
  LineValueLabelBox = (color, mode, depth);

- **LineValueLabelStyle**
  LineValueLabelStyle = labelposition1, labelposition2, ...

- **PlotArea**
  PlotArea = (xlocation, ylocation, width, height);

- **StripLayout**
  StripLayout = (NumSlots, InitialFill, MaxFill, UndefinedString);

- **TopLabels**
  TopLabels = "Label1", "Label2", ...

- **TopScale**
  TopScale = (MinValue, MaxValue, StepValue);

- **TopScroll**
  TopScroll = (ScrollMin, ScrollMax);

- **Update**
  Update;

**Time Chart**

**Generally Supported Parameters**

- **ActiveLabels[1-50]**
  ActiveLabels[1-50] = ("Label1", "URL1", "Target1"), ...
  See also Chapter 6, Active Labels and Drill-Down

- **Axis**
  See Chapter 6, Axis Modifications for various parameters available

- **Color Table**
  See Chapter 8, Color for application of color attributes

- **Background**
  Background = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor);

- **BackgroundFillPattern**
  BackgroundFillPattern = (type, color1, color2, imageURL), ...

- **ChartElementSpacing**
  ChartElementSpacing = spacing;

- **Grid**
  Grid = (LineColor1, bgColor1, borderColor1, bgColor1, ImageFormat1), ...
Legend
Legend  =  ("Label", Color, "FontName", FontSize, Angle, interiorAlignment);

Note Sets
NoteSets = ("Name1", Justify1), ("Name2", Justify2), ...;
See also Chapter 6, Notes, or Annotations

Title
See Chapter 6,
Labels for various parameters available.

TitleBox
See Chapter 6, Regions, or Boxes for various parameters available.

Specifically Supported Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataAxis</td>
<td>DataAxis = (XAxis1, YAxis1), (XAxis2, YAxis2), ...;</td>
</tr>
<tr>
<td>DataSets</td>
<td>DataSets = (DataSetName1, Color1, BorderType1, BorderWidth1, &quot;ImageURL&quot;, ImageFormat1, BorderColor1), ...;</td>
</tr>
<tr>
<td>DataSet[n]</td>
<td>DataSet[1-50] = a, b, c, ...;</td>
</tr>
<tr>
<td>PlotArea</td>
<td>PlotArea = (xlocation, ylocation, width, height);</td>
</tr>
<tr>
<td>TaskColorTable[n]</td>
<td>TaskColorTable[1-50] = color1, color2,...;</td>
</tr>
<tr>
<td>TaskHeight</td>
<td>TaskHeight = value;</td>
</tr>
<tr>
<td>UniqueTaskColors</td>
<td>UniqueTaskColors  = ON</td>
</tr>
</tbody>
</table>

X-Y Chart

Generally Supported Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveLabels[1-50]</td>
<td>ActiveLabels[1-50] = (&quot;Label1&quot;, &quot;URL1&quot;, &quot;Target1&quot;), ...;</td>
</tr>
<tr>
<td>Axis</td>
<td>See also Chapter 6, Active Labels and Drill-Down</td>
</tr>
<tr>
<td>Axis Thickness</td>
<td>AxisThickness = 15;</td>
</tr>
<tr>
<td>BuildAnimation Enabled</td>
<td>BuildAnimationEnabled = ON</td>
</tr>
<tr>
<td>Color Table</td>
<td>See Chapter 8, Color for application of color attributes</td>
</tr>
<tr>
<td>Background</td>
<td>Background = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor);</td>
</tr>
<tr>
<td>BackgroundFillPattern</td>
<td>BackgroundFillPattern = (type, color1, color2, imageURL), ...;</td>
</tr>
<tr>
<td>ChartElementSpacing</td>
<td>ChartElementSpacing = spacing;</td>
</tr>
<tr>
<td>Grid</td>
<td>Grid = (LineColor1, bgColor1, borderColor1, bgImage1, ImageFormat1), ...;</td>
</tr>
<tr>
<td>Legend</td>
<td>Legend  =  (&quot;Label&quot;, Color, &quot;FontName&quot;, FontSize, Angle, interiorAlignment);</td>
</tr>
<tr>
<td>Note Sets</td>
<td>NoteSets = (&quot;Name1&quot;, Justify1), (&quot;Name2&quot;, Justify2), ...;</td>
</tr>
<tr>
<td>Title</td>
<td>See Chapter 6, Notes, or Annotations</td>
</tr>
</tbody>
</table>
| TitleBox                  | See Chapter 6,
Labels, for various parameters available. |

Specifically Supported Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddDataPoint</td>
<td>AddDataPoint = (&quot;Name&quot;, X, Y, Z, &quot;Label&quot;, &quot;URL&quot;, &quot;Target&quot;), ...;</td>
</tr>
<tr>
<td>DwellOffset</td>
<td>DwellOffset = size;</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Line3DDepth</td>
<td>Line3DDepth = depth;</td>
</tr>
<tr>
<td>LineAxis</td>
<td>LineAxis = (XAxis1, YAxis1), (XAxis2, YAxis2), ...;</td>
</tr>
<tr>
<td>LineColorTable[n]</td>
<td>LineColorTable[1-50] = color1, color2...;</td>
</tr>
<tr>
<td>LineFillPattern</td>
<td>LineFillPattern = (type, color1, color2, imageURL), ...;</td>
</tr>
<tr>
<td>LineSets</td>
<td>LineSets = (Name1, SymColor1), (Name2, SymColor2), ...;</td>
</tr>
<tr>
<td>LineSet[n]</td>
<td>LineSet[1-50] = y1, y2, y3, ...;</td>
</tr>
<tr>
<td>LineStyle</td>
<td>LineStyle = (LineType, LineWidth, Color, FillColor, LineType, FillType), ...;</td>
</tr>
<tr>
<td>LineSymbolSpotlights</td>
<td>LineSymbolSpotlights = (start, stop, center, centeroffsetx, centeroffsety, focusoffsetx, focusoffsety, radius), ...;</td>
</tr>
<tr>
<td>LineValueLabelBox</td>
<td>LineValueLabelBox = (color, mode, depth);</td>
</tr>
<tr>
<td>LineValueLabelStyle</td>
<td>LineValueLabelStyle = labelposition1, labelposition2, ...;</td>
</tr>
<tr>
<td>LineWidth</td>
<td>LineWidth = PercentDepth;</td>
</tr>
<tr>
<td>PlotArea</td>
<td>PlotArea = (xlocation, ylocation, width, height);</td>
</tr>
</tbody>
</table>
6. CDL Parameters Arranged by Function

Once you have mastered the anatomy of a chart, you may wish to reference CDL parameters by their function. This directory serves as such a reference, with more direct commentary on particular functionality that you will find elsewhere in this Guide.

**Active Labels and Drill-Down**

All Visual Mining charting software, except sometimes images generated from NetCharts Server or NetCharts Pro, support the display of informational labels, called dwell or active labels, which are seen whenever the mouse dwells over a specified data value or label. For example, on a Bar chart, the value of an individual bar can be displayed when the mouse cursor hovers over a bar for a short period of time. Alternatively, a user prompt can be displayed when the mouse hovers over a legend item. You can see numerous such examples on our web site, at this URL:

```
www.visualmining.com/examples/javaexamples/index.html
```

If one clicks the mouse while an active label is displayed, the active label may serve to navigate to a predefined URL. This URL itself can serve any of three purposes:

- To replace the current HTML document with any other HTML document
- To alter the display of any named frame or window within the browser.
- To load new chart parameters from a parameter file.

This capability is often called drill-down, and is extremely flexible, in that any chart may thus serve as a graphical interface to additional information, be it visual or textual, that is accessible from another document or CGI script.

If the `DwellLabel` parameter is also defined, then a label will automatically be displayed whenever the mouse cursor dwells over a given data value. The `ActiveLabels` parameter defined for a dwell label parameter then specifies the format for each label, and not the text value.

**Parameters Involved, Alphabetically**

- **ActiveClicks**
  
  ```
  ActiveClicks = Number;
  ```

- **ActiveLabels**

  ```
  ActiveLabels = ("Label1", "URL1", "Target1"), ...;
  ```

- **ActiveLabels[1-50]**

  ```
  ActiveLabels[1-50] = ("Label", "URL", "Target"), ...;
  ```

- **BackgroundActiveLabel**

  ```
  BackgroundActiveLabel = ("Label", "URL", "Target");
  ```

- **BarActiveLabels**

  ```
  BarActiveLabels = ("Label", "URL", "Target"), ...;
  ```

- **BottomActiveLabels**

  ```
  BottomActiveLabels = ("Label", "URL", "Target");
  ```

- **BottomAxisTitleActiveLabel**

  ```
  BottomAxisTitleActiveLabel = ("Label", "URL", "Target");
  ```

- **BoxActiveLabels**

  ```
  BoxActiveLabels = ("Label", "URL", "Target"), ...;
  ```

- **DataPointActiveLabels[1-50]**

  ```
  DataPointActiveLabels[1-50] = ("Label", "URL", "Target");
  ```

- **DwellAnimationHighlightBorderStyle**

  ```
  DwellAnimationHighlightBorderStyle = (lineType, LineWidth, lineColor);
  ```

- **DwellAnimationHighlightFill**

  ```
  DwellAnimationHighlightFill = (color);
  ```
Axis Modifications

Most Visual Mining charts support the definition and display of one or more axes, depending on the specific chart type. Currently, the following standard axes are defined for most charts, for top, bottom, left, and right axes.

Generally, the bottom and left axes are used by default to map the X and Y data values, although in the Time chart, for example, the top axis is the default X axis.

Parameters Involved, Alphabetically

AutoscalePad  AutoscalePad = pad;
AxisThickness  AxisThickness = thickness;
BottomActiveLabels  BottomActiveLabels = ("Label", "URL", "Target");
BottomAxesGaps  BottomAxesGaps = value, value, ...;
BottomAxesLayout  BottomAxesLayout = value, value, ...;
<table>
<thead>
<tr>
<th>Function</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>BottomAxisTitle</td>
<td>BottomAxisTitle = (Label, Color, FontName, FontSize, Angle,</td>
</tr>
<tr>
<td></td>
<td>interiorAlignment, exteriorAlignment)</td>
</tr>
<tr>
<td>BottomAxisTitleActiveLabel</td>
<td>BottomAxisTitleActiveLabel = (&quot;Label&quot;, &quot;URL&quot;, &quot;Target&quot;)</td>
</tr>
<tr>
<td>BottomAxisTitleBox</td>
<td>BottomAxisTitleBox = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;,</td>
</tr>
<tr>
<td></td>
<td>ImageFormat, BorderColor)</td>
</tr>
<tr>
<td>BottomColor</td>
<td>BottomColor = Color;</td>
</tr>
<tr>
<td>BottomDrawMinorTics</td>
<td>BottomDrawMinorTics = ON</td>
</tr>
<tr>
<td>BottomFormat</td>
<td>BottomFormat = (FormatType, &quot;FormatExpr&quot;, &quot;TimeBase&quot;, &quot;TimeUnit&quot;)</td>
</tr>
<tr>
<td>BottomLabels</td>
<td>BottomLabels = &quot;Label1&quot;, &quot;Label2&quot;, ...;</td>
</tr>
<tr>
<td>BottomMajorTicColor</td>
<td>BottomMajorTicColor = Color;</td>
</tr>
<tr>
<td>BottomMinorTicColor</td>
<td>BottomMinorTicColor = Color;</td>
</tr>
<tr>
<td>BottomMargins</td>
<td>BottomMargins = (LeftSideMargin, RightSideMargin);</td>
</tr>
<tr>
<td>BottomScale</td>
<td>BottomScale = (MinValue, MaxValue, StepValue);</td>
</tr>
<tr>
<td>BottomScaleMode</td>
<td>BottomScaleMode = (mode, logBase), (mode, logBase), ...;</td>
</tr>
<tr>
<td>BottomScaleSet</td>
<td>BottomScaleSet = (MinValue, MaxValue, StepValue, Percentage);</td>
</tr>
<tr>
<td>BottomScroll</td>
<td>BottomScroll = (ScrollMin, ScrollMax);</td>
</tr>
<tr>
<td>BottomTics</td>
<td>BottomTics = (&quot;Mode&quot;, Color, &quot;FontName&quot;, FontSize, Angle,</td>
</tr>
<tr>
<td></td>
<td>interiorAlignment);</td>
</tr>
<tr>
<td>BottomTicLayout</td>
<td>BottomTicLayout = (Mode, SkipCount, StaggerLevels);</td>
</tr>
<tr>
<td>BottomTicLocations</td>
<td>BottomTicLocations = value, value, value, ...;</td>
</tr>
<tr>
<td>BottomTitle</td>
<td>BottomTitle = (&quot;Label&quot;, Color, &quot;FontName&quot;, FontSize, Angle,</td>
</tr>
<tr>
<td></td>
<td>interiorAlignment, exteriorAlignment);</td>
</tr>
<tr>
<td>BottomTitleActiveLabel</td>
<td>BottomTitleActiveLabel = (&quot;Label&quot;, &quot;URL&quot;, &quot;Target&quot;);</td>
</tr>
<tr>
<td>LeftActiveLabels</td>
<td>LeftActiveLabels = (&quot;Label&quot;, &quot;URL&quot;, &quot;Target&quot;);</td>
</tr>
<tr>
<td>LeftAxesGaps</td>
<td>LeftAxesGaps = value, value, ...;</td>
</tr>
<tr>
<td>LeftAxesLayout</td>
<td>LeftAxesLayout = value, value, ...;</td>
</tr>
<tr>
<td>LeftAxisTitle</td>
<td>LeftAxisTitle = (Label, Color, FontName, FontSize, Angle,</td>
</tr>
<tr>
<td></td>
<td>interiorAlignment, exteriorAlignment);</td>
</tr>
<tr>
<td>LeftAxisTitleActiveLabel</td>
<td>LeftAxisTitleActiveLabel = (&quot;Label&quot;, &quot;URL&quot;, &quot;Target&quot;);</td>
</tr>
<tr>
<td>LeftAxisTitleBox</td>
<td>LeftAxisTitleBox = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;,</td>
</tr>
<tr>
<td></td>
<td>ImageFormat, BorderColor)</td>
</tr>
<tr>
<td>LeftColor</td>
<td>LeftColor = Color;</td>
</tr>
<tr>
<td>LeftDrawMinorTics</td>
<td>LeftDrawMinorTics = ON</td>
</tr>
<tr>
<td>LeftFormat</td>
<td>LeftFormat = (FormatType, &quot;FormatExpr&quot;, &quot;TimeBase&quot;, &quot;TimeUnit&quot;)</td>
</tr>
<tr>
<td>LeftLabels</td>
<td>LeftLabels = &quot;Label1&quot;, &quot;Label2&quot;, ...;</td>
</tr>
<tr>
<td>LeftMajorTicColor</td>
<td>LeftMajorTicColor = Color;</td>
</tr>
<tr>
<td>LeftMinorTicColor</td>
<td>LeftMinorTicColor = Color;</td>
</tr>
<tr>
<td>LeftMargins</td>
<td>LeftMargins = (BottomSideMargin, TopSideMargin);</td>
</tr>
<tr>
<td>LeftScale</td>
<td>LeftScale = (MinValue, MaxValue, StepValue);</td>
</tr>
<tr>
<td>LeftScaleMode</td>
<td>LeftScaleMode = (mode, logBase), (mode, logBase), ...;</td>
</tr>
<tr>
<td>LeftScaleSet</td>
<td>LeftScaleSet = (MinValue, MaxValue, StepValue, Percentage);</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>LeftScroll</td>
<td><code>LeftScroll = (ScrollMin, ScrollMax)</code>;</td>
</tr>
<tr>
<td>LeftTics</td>
<td>`LeftTics = &quot;Mode&quot;, Color, &quot;FontName&quot;, FontSize, Angle, interiorAlignment&quot;;</td>
</tr>
<tr>
<td>LeftTicLayout</td>
<td><code>LeftTicLayout = (Mode, SkipCount, StaggerLevels)</code>;</td>
</tr>
<tr>
<td>LeftTicLocations</td>
<td><code>LeftTicLocations = value, value, value, ...;</code></td>
</tr>
<tr>
<td>LeftTitle</td>
<td>`LeftTitle = &quot;Label&quot;, Color, &quot;FontName&quot;, FontSize, Angle, interiorAlignment, exteriorAlignment&quot;;</td>
</tr>
<tr>
<td>LeftTitleActiveLabel</td>
<td><code>LeftTitleActiveLabel = &quot;Label&quot;, &quot;URL&quot;, &quot;Target&quot;;</code></td>
</tr>
<tr>
<td>RightActiveLabels</td>
<td><code>RightActiveLabels = &quot;Label&quot;, &quot;URL&quot;, &quot;Target&quot;;</code></td>
</tr>
<tr>
<td>RightAxesGaps</td>
<td><code>RightAxesGaps = value, value, ...;</code></td>
</tr>
<tr>
<td>RightAxesLayout</td>
<td><code>RightAxesLayout = value, value, ...;</code></td>
</tr>
<tr>
<td>RightAxisTitle</td>
<td>`RightAxisTitle = &quot;Label&quot;, Color, &quot;FontName&quot;, FontSize, Angle, interiorAlignment, exteriorAlignment&quot;;</td>
</tr>
<tr>
<td>RightAxisTitleActiveLabel</td>
<td><code>RightAxisTitleActiveLabel = &quot;Label&quot;, &quot;URL&quot;, &quot;Target&quot;;</code></td>
</tr>
<tr>
<td>RightAxisTitleBox</td>
<td><code>RightAxisTitleBox = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor)</code></td>
</tr>
<tr>
<td>RightColor</td>
<td><code>RightColor = Color;</code></td>
</tr>
<tr>
<td>RightDrawMinorTics</td>
<td>`RightDrawMinorTics = ON</td>
</tr>
<tr>
<td>RightFormat</td>
<td><code>RightFormat = (FormatType, &quot;FormatExpr&quot;, &quot;TimeBase&quot;, &quot;TimeUnit&quot;)</code></td>
</tr>
<tr>
<td>RightLabels</td>
<td><code>RightLabels = &quot;Label1&quot;, &quot;Label2&quot;, ...;</code></td>
</tr>
<tr>
<td>RightMajorTicColor</td>
<td><code>RightMajorTicColor = Color;</code></td>
</tr>
<tr>
<td>RightMinorTicColor</td>
<td><code>RightMinorTicColor = Color;</code></td>
</tr>
<tr>
<td>RightMargins</td>
<td><code>RightMargins = (TopSideMargin, BottomSideMargin);</code></td>
</tr>
<tr>
<td>RightScale</td>
<td><code>RightScale = (MinValue, MaxValue, StepValue);</code></td>
</tr>
<tr>
<td>RightScaleMode</td>
<td><code>RightScaleMode = (mode.logBase), (mode.logBase), ...;</code></td>
</tr>
<tr>
<td>RightScaleSet</td>
<td><code>RightScaleSet = (MinValue, MaxValue, StepValue, Percentage);</code></td>
</tr>
<tr>
<td>RightScroll</td>
<td><code>RightScroll = (ScrollMin, ScrollMax)</code>;</td>
</tr>
<tr>
<td>RightTics</td>
<td>`RightTics = &quot;Mode&quot;, Color, &quot;FontName&quot;, FontSize, Angle, interiorAlignment&quot;;</td>
</tr>
<tr>
<td>RightTicLayout</td>
<td><code>RightTicLayout = (Mode, SkipCount, StaggerLevels)</code></td>
</tr>
<tr>
<td>RightTicLocations</td>
<td><code>RightTicLocations = value, value, value, ...;</code></td>
</tr>
<tr>
<td>RightTitle</td>
<td>`RightTitle = &quot;Label&quot;, Color, &quot;FontName&quot;, FontSize, Angle, interiorAlignment, exteriorAlignment&quot;;</td>
</tr>
<tr>
<td>RightTitleActiveLabel</td>
<td><code>RightTitleActiveLabel = &quot;Label&quot;, &quot;URL&quot;, &quot;Target&quot;;</code></td>
</tr>
<tr>
<td>TopActiveLabels</td>
<td><code>TopActiveLabels = &quot;Label&quot;, &quot;URL&quot;, &quot;Target&quot;;</code></td>
</tr>
<tr>
<td>TopAxesGaps</td>
<td><code>TopAxesGaps = value, value, ...;</code></td>
</tr>
<tr>
<td>TopAxesLayout</td>
<td><code>TopAxesLayout = value, value, ...;</code></td>
</tr>
<tr>
<td>TopAxisTitle</td>
<td>`TopAxisTitle = &quot;Label&quot;, Color, &quot;FontName&quot;, FontSize, Angle, interiorAlignment, exteriorAlignment&quot;;</td>
</tr>
<tr>
<td>TopAxisTitleActiveLabel</td>
<td><code>TopAxisTitleActiveLabel = &quot;Label&quot;, &quot;URL&quot;, &quot;Target&quot;;</code></td>
</tr>
<tr>
<td>TopAxisTitleBox</td>
<td><code>TopAxisTitleBox = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor)</code></td>
</tr>
</tbody>
</table>
CDL Parameters Arranged by Function

TopColor
TopColor = Color;

TopDrawMinorTics
TopDrawMinorTics = ON, OFF;

TopFormat
TopFormat = (FormatType, "FormatExpr", "TimeBase", "TimeUnit");

TopLabels
TopLabels = "Label1", "Label2", ...;

TopMajorTicColor
TopMajorTicColor = Color;

TopMinorTicColor
TopMinorTicColor = Color;

TopMargins
TopMargins = (LeftSideMargin, RightSideMargin);

TopScale
TopScale = (MinValue, MaxValue, StepValue);

TopScaleMode
TopScaleMode = (mode, logBase), (mode, logBase), ...

TopScaleSet
TopScaleSet = (MinValue, MaxValue, StepValue, Percentage);

TopScroll
TopScroll = (ScrollMin, ScrollMax);

TopTics
TopTics = ("Mode", Color, "FontName", FontSize, Angle,
interiorAlignment, backgroundColor, rotationPoint);

TopTicLayout
TopTicLayout = (Mode, SkipCount, StaggerLevels);

TopTicLocations
TopTicLocations = value, value, value, ...

TopTitle
TopTitle = ("Label", Color, "FontName", FontSize, Angle,
interiorAlignment, exteriorAlignment);

TopTitleActiveLabel
TopTitleActiveLabel = ("Label", "URL", "Target");

Grids

All Visual Mining charts, except for the Dial chart, Diagram chart, and Pie chart, support the display of one or more optional grids behind the data. The grid, if used, forms a background behind the chart, making measurement against the axes easier when reading it.

With Visual Mining products, you can specify several grid characteristics, including different colored and styled grid lines, grid background, and grid depth. Additionally, you can define more than one grid per chart, which is useful when your chart requires independent Y axes.

In additions to lines and bars, you can also use images in the grid. The two charts below indicate some of the variety one can achieve using the grid parameters.

![Bar Chart with 3-D Grid with Image](image1)

![Bubble Chart Using Double Grid and Double Y Axes](image2)
Parameters Involved, Alphabetically

Grid is the fundamental parameter for setting up grids, and you must use it first when setting up other grid parameters.

```
Grid = (LineColor1, bgColor1, borderColor1, bgImage1, ImageFormat1), ...
```

```
Grid3DDepth = depth;
```

```
GridAxis = (XAxis1, YAxis1), ...
```

```
GridLine = (LineType,LineStyle,LineWidth), ...
```

Labels

Label definitions are used extensively throughout Visual Mining charts in order to display titles, legends, axis tics, data labels, etc. In most cases, labels use a standard vector, or “tuple,” to define the text and its appearance. Thus, you are able to control:

- The text itself, or in the case of axis tic labels, only whether the label is shown or not
- Color
- Font name and attributes
- Font size
- Text angle

Font Names and Attributes

The font names available to a NetChart depend on the Operating System and the Java Virtual Machine being used to run the software. For example, the fonts available to NetCharts running with JVM 1.2 on Solaris will be different than the fonts available running with JVM 1.4 on Windows XP.

Font names can be augmented with additional font style information. Adding "Plain", "Bold" or "Italic" to the font name modifies the style as specified. For example "Courier Bold Italic" is a valid font name specification. By default NetCharts chooses a BOLD style for the specified font so "TimesRoman" is equivalent to "TimesRoman Bold". To get a standard version of a font, add "Plain" to the name.

Font names can also be augmented with "underline", "overline", "linethrough", "ascent=N", "descent=N","leading=N" and "maxLineAdvance=N".

Any combination of style modifiers is allowed. For example

```
TimesRoman Plain Italic underline overline linethrough ascent=5 descent=0 leading=-5
```

is a valid font specification which uses a nonbold italic type, draws a line above, below and through the text, and controls the amount of space allocated for character ascents, descents and the space between consecutive lines.

maxLineAdvance controls the maximum length of a string before NetCharts breaks to a new line.
Parameters Involved, Alphabetically

The following parameters follow the label standard format, allowing you to customize your text.

```
BottomAxisTitle = (Label, Color, FontName, FontSize, Angle, interiorAlignment, exteriorAlignment);
BottomTics = ("Mode", Color, "FontName", FontSize, Angle, interiorAlignment, backgroundColor, rotationPoint);
BottomTitle = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment, exteriorAlignment);
DwellLabel = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment);
Footer = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment, exteriorAlignment);
Header = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment, exteriorAlignment, extend);
GridBlockLabel = ("mode", color, "font name", font size, Angle, interiorAlignment);
LeftAxisTitle = (Label, Color, FontName, FontSize, Angle, interiorAlignment, exteriorAlignment);
LeftTics = ("Mode", Color, "FontName", FontSize, Angle, interiorAlignment, backgroundColor, rotationPoint);
LeftTitle = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment, exteriorAlignment, extend);
Legend = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment);
RightAxisTitle = (Label, Color, FontName, FontSize, Angle, interiorAlignment, exteriorAlignment);
RightTics = ("Mode", Color, "FontName", FontSize, Angle, interiorAlignment, backgroundColor, rotationPoint);
RightTitle = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment, exteriorAlignment, extend);
TopAxisTitle = (Label, Color, FontName, FontSize, Angle, interiorAlignment, exteriorAlignment);
TopTics = ("Mode", Color, "FontName", FontSize, Angle, interiorAlignment, backgroundColor, rotationPoint);
TopTitle = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment, exteriorAlignment);
```

Legends

Legends, which are text or text combined with symbols, are often used as titles or as interpretive keys to the chart data. Legends are easily modified to suit any number of potential uses beyond titling a data set.

All Visual Mining charts, except for the Diagram chart, support the display of a legend anywhere on the chart, including inside the grid. The entries for the legend can be fully specified, or can be loaded automatically using the names and colors assigned to all data sets being displayed.

The fundamental parameter for setting up legends is `Legend`. You must use this one if you wish to adjust any other legend-related parameters.

Parameters Involved, Alphabetically

The following parameters follow the label standard format, allowing you to customize your text.
DataLegend

\[ \text{DataLegend} = \text{ON} | \text{OFF}; \]

DataLegendGrid

\[ \text{DataLegendGrid} = (\text{lineColor}, \text{bgColor}, \text{borderColor}, \text{bgImage}, \text{bgImageType}); \]

DataLegendGridLine

\[ \text{DataLegendGridLine} = (\text{lineType1}, \text{lineStyle1}, \text{lineWidth1}); \]

Legend

\[ \text{Legend} = (\"Label\", \text{Color}, \"FontName\", \text{FontSize}, \text{Angle}, \text{interiorAlignment}); \]

LegendActiveLabels

\[ \text{LegendActiveLabels} = (\"Label\", \"URL\", \"Target\"), \ldots; \]

LegendAxis

\[ \text{LegendAxis} = (\text{XAxis}, \text{Yaxis}); \]

LegendBox

\[ \text{LegendBox} = (\text{Color}, \text{BorderType}, \text{BorderWidth}, \"ImageURL\", \text{ImageFormat}, \text{BorderColor}); \]

LegendBoxSize

\[ \text{LegendBoxSize} = (\text{MaxWidth}, \text{MaxHeight}); \]

LegendItems

\[ \text{LegendItems} = (\"Label1\", \text{Color1}, \text{SymType1}, \text{SymSize1}, \text{SymStyle1}, \text{LineType1}, \text{LineWidth1}, \text{LineColor1}, \text{Patternfill}, \text{Patternforeground}, \text{Patternbackground}, \text{Image}, \text{Shadowwidth}), \ldots; \]

LegendItemBox

\[ \text{LegendItemBox} = (\text{lineType}, \text{LineWidth}, \text{lineColor}); \]

LegendLayout

\[ \text{LegendLayout} = (\text{Type}, \text{Location}, \text{X}, \text{Y}, \text{Justify}, \text{Columns}, \text{ItemOrder}); \]

LegendSymbolPosition

\[ \text{LegendSymbolPosition} = \text{LEFT} | \text{RIGHT}; \]

**Notes, or Annotations**

When designing a chart, you can place text annotations, or notes, anywhere on the graph, both inside the chart and alongside it. Notes may have optional arrows with up to three bends in them, and three endpoint shapes. Notes may also become targets for drill-downs to other charts and data (see Active Labels, above).

Using Notes, items of interest can be easily identified, and critical regions can be highlighted. Notes may also be used to turn charts into illustrations.

Notes combine the functions of Label and Region, since they are text on various sorts of box backgrounds. The fundamental parameter for setting up notes is **NoteSets**.

**Parameters Involved, Alphabetically**

**NoteActiveLabels**

\[ \text{NoteActiveLabels}[1-20] = (\"Label\", \"URL\", \"Target\"), \ldots; \]

**NoteArrow**

\[ \text{NoteArrow} = (\text{LineType1}, \text{LineWidth1}, \text{LineColor1}, \text{ArrowType1}, \text{ArrowStyle1}), (\text{LineType2}, \text{LineWidth2}, \text{LineColor2}, \text{ArrowType2}, \text{ArrowStyle2}), \ldots; \]

**NoteAxis**

\[ \text{NoteAxis} = (\text{XAxis1}, \text{YAxis1}), (\text{XAxis2}, \text{YAxis2}), \ldots; \]

**NoteBox**

\[ \text{NoteBox} = (\text{Color1}, \text{BorderType1}, \text{BorderWidth1}, \text{ImageURL1}, \text{ImageFormat1}), \ldots; \]

**NoteLabel**

\[ \text{NoteLabel} = (\"Label1\", \text{Color1}, \"FontName1\", \text{FontSize1}, \text{Angle1}, \text{interiorAlignment1}), \ldots; \]

**NoteSets**

\[ \text{NoteSets} = (\"Name1\", \text{Justify1}), (\"Name2\", \text{Justify2}), \ldots; \]

**NoteSet[n]**

\[ \text{NoteSet1} = (\"Text1\", \text{X}, \text{Y}, \text{X1}, \text{Y1}, \text{X2}, \text{Y2}, \text{X3}, \text{Y3}), \ldots; \]
Regions, or Boxes

Region definitions are used extensively throughout Visual Mining charts in displaying titles, legends, data labels, dwell labels, backgrounds, and notes. A region is displayed as a rectangular box with a specified color and border type. In most cases, these boxes are optional and need not appear at all.

Stylistically, we recommend that you match the chart background (as opposed to its grid) to the color of the web page you are using, so that it blends in and does not distract the viewer from the data. Similarly, we recommend that you use boxes for titles, notes, and legends very sparingly, as these can also be distracting, and do not add to the meaning of the graph. Only dwell labels really require some contrasting color or edge décor to set them off when they pop up.

All region parameters use a basic group, or “tuple,” of attributes:

- Main color
- Border type (raised, inset, shadowed, box, or none)
- Border width, in pixels
- Image URL, which indicates an image file that serves at the box color
- Image format (tiled, centered, sized to fit)
- Border color

Additionally some regions support a corner style and corner color. Corner styles are specified in a clockwise fashion starting in the upper left - Top Left, Top Right, Bottom Right, Bottom Left. Legal values are SQUARE, SNIP and ROUND. The default is SQUARE. The CornerColor attribute specifies the color to fill corners of a region when the CornerStyle is SNIP or ROUND.

Parameters Involved, Alphabetically

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background</td>
<td>Background = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);</td>
</tr>
<tr>
<td>BottomAxisTitleBox</td>
<td>BottomAxisTitleBox = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);</td>
</tr>
<tr>
<td>DwellBox</td>
<td>DwellBox = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);</td>
</tr>
<tr>
<td>FooterBox</td>
<td>FooterBox = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);</td>
</tr>
<tr>
<td>HeaderBox</td>
<td>HeaderBox = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);</td>
</tr>
<tr>
<td>LeftAxisTitleBox</td>
<td>LeftAxisTitleBox = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);</td>
</tr>
<tr>
<td>LeftTitleBox</td>
<td>LeftTitleBox = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);</td>
</tr>
<tr>
<td>LegendBox</td>
<td>LegendBox = (Color, BorderType, BorderWidth, &quot;ImageURL&quot;, ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);</td>
</tr>
</tbody>
</table>
NoteBox

NoteBox = (Color1, BorderType1, BorderWidth1, ImageURL1, ImageFormat1, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor), ...

PieBackgrounds

PieBackgrounds = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor);

RightAxisTitleBox

RightAxisTitleBox = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);

RightTitleBox

RightTitleBox = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);

TopAxisTitleBox

TopAxisTitleBox = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);
7. CDL Parameter Definitions

As was discussed in Chapter 2, CDL parameters may either be used in the standard HTML format for applets, or within the NFPParamScript format.

As in the rest of the guide, the attribute list for each CDL parameter points to common attributes, which are defined in Chapter 4, when the attribute name is in italic. When you see an attribute listed in italic, it means that you should look to that section for specific details about the attribute. If you recognize a term from the list of attributes-in-common, but it is not shown in italic, that means that there is some parameter-specific information about the attribute, so the description is provided with that element.

In the Examples for a parameter-specific attribute, the attribute is shown in bold, to help you locate it. In actual code, attributes would not be bolded.

**ActiveClicks**

ActiveClicks = Number;

In Active Label processing on NetCharts Applets, use **ActiveClicks** to specify a number of mouse clicks needed to activate the label. The user can then click the mouse while an active label is displayed, resulting in a predefined URL being executed. This URL can serve any of three purposes:

- To replace the current HTML document with any other HTML document
- To alter the display of any named frame or window within the browser
- To load new chart parameters from a parameter file

This capability is sometimes referred to as “drill-down.”

**Used in These Charts**

All

**Attributes**

Number

**Number**

Number is the number of clicks of the mouse that the user must make to activate a label.

**Example:**

ActiveClicks = 2;

**Values**

A whole number, preferably one to three.

**Default**

1
ActiveLabels

ActiveLabels[N] = ("Label1", "URL1", "Target1"), ...;

The ActiveLabels parameter is the basis for significant interactivity within the Visual Mining charting applications. For each chart, it can provide a set of one or more labels that respond to mouse dwell or mouse clicks.

If the DwellLabel parameter is also defined, then a label will automatically be displayed whenever the mouse cursor dwells over a given data value. The ActiveLabels parameter defined for a dwell label parameter then specifies the format of each label, and not the text value.

All of the other ActiveLabels attributes are optional and may be used to control the dwell label for one or more data values, axis labels, legend labels or titles. In most charts, the ActiveLabels parameter is specified using a numeric ID (such as ActiveLabels1 or ActiveLabels5) to indicate which data set is being referenced. Each tuple defined in the ActiveLabels parameter is matched with the corresponding data value defined in the DataSet or equivalent parameter. If too many active labels are defined, the extra tuples are ignored. If too few are defined, the remaining data values will use the default dwell label generated.

All of the other ActiveLabels attributes are optional and may be used to control the dwell label for one or more data values, axis labels, legend labels or titles. In most charts, the ActiveLabels parameter is specified using a numeric ID (such as ActiveLabels1 or ActiveLabels5) to indicate which data set is being referenced. Each grouped, parenthesized set, or “tuple,” defined in the ActiveLabels parameter is matched with the corresponding data value defined in the DataSet or equivalent parameter. If too many active labels are defined, the extra tuples are ignored. If too few are defined, the remaining data values will use the default dwell label generated.

Used in These Charts

All

Example:

ActiveLabels = ("", "Barchart9Mon.html", "InfoFrame"),
("", "Barchart9Tue.html"),
("Barchart9Wed.html"),
(OUTLINE", "Barchart9Thu.html"),
(OUTLINE", "Barchart9Fri.html");

Attributes

<table>
<thead>
<tr>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
</table>

ActiveLabels[n]

ActiveLabels[1-50] = ("Label", "URL", “Target”), ...;

In most charts, the ActiveLabels parameter is specified using a numeric ID (such as ActiveLabels1 or ActiveLabels5) to indicate which data set is being referenced. Each such set defined in the ActiveLabels parameter is matched with the corresponding data value defined in the DataSet or
equivalent parameter. If too many active labels are defined, the extra sets are ignored. If too few are defined, the remaining data values will use the default dwell label generated.

**Used in These Charts**

All

**Example:**

ActiveLabels2 = ("One Hundred", "Barchart9Mon.html", "InfoFrame"),
("One Hundred\nTwenty Five", "noinfo.html"),
("Two Hundred\nForty Five", "Barchart9Wed.html"),
("OUTLINE", "Barchart9Thu.html"),
("OUTLINE", "Barchart9Fri.html");

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
</table>

**AddDataPoint**

AddDataPoint = ("Name", X, Y, Z, "Label", "URL", "Target"), ...;

AddDataPoint is used in bubble charts, but is only used if you are plotting dynamically. The attributes correspond with data points named in the BubbleSets parameter. AddDataPoint must be the only command sent that is followed by the Update command. Using AddDataPoint allows the bubble chart to be updated rapidly without refreshing the screen or reconfiguring the layout. The Label, URL, and Target attributes are optional, and if specified, are used to define an ActiveLabel for the given data point. If NULL is substituted for either or both of the (X,Y) pair values, the point will not be drawn.

**Used in These Charts**

Bubble, X-Y

**Example:**

AddDataPoint = ("R1", 320, 199, 3.85),
("R2", 445.8, 622, 2.21, "Horizon", "horizon.html", "_self");

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>Name</th>
<th>Target</th>
<th>URL</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
</table>

**X, Y, and Z**

X, Y, and Z represent the dynamically updated coordinates of the Named data points vector, along the associated X, Y, and Z axes.

**Example:**

AddDataPoint = ("R1", 320, 199, 3.85);
Common CDL Attributes

Values
Real numbers

Default
None

AppendDataSet[n]

AppendDataSet[1-50] = y1, y2, y3, ...;

AppendDataSet defines a list of Y values for each data set defined by the DataSets parameter. The data is appended onto the data queue for the specified dataSet. Data is extracted from the queue when an Update statement is passed. If the value “null” is substituted for a y value, the symbol is not drawn for that slot.

Used in These Charts
Strip

Example:

AppendDataSet1 = 35, 38, 42, 41, 40, 37.5, 36.125, 35, 38;

Attributes

Y

Y

Y represent the dynamically updated Y-coordinates of the data points for the specific data set.

Example:

AppendDataSet1 = 35, 38, 42, 41, 40, 37.5, 36.125, 35, 38;

Values
Real numbers
null The symbol is not drawn for that position

Default
None

AppendPolyDataToActiveLabels

AppendPolyDataToActiveLabels = ON | OFF;
If **AppendPolyDataToActiveLabels** is ON, then Diagram Polygon Active Labels will automatically contain both the polygon label and the current data value associated with the polygon. If the value is OFF, the Diagram Polygon Active Labels will only contain the polygon label. The default value is ON.

**Used in These Charts**
Diagram

**Example:**

```plaintext
AppendPolyDataToActiveLabels = OFF;
```

**Attributes**

**Mode**

**AntiAlias**

**AntiAlias = mode;**

NetCharts includes support for anti-aliased fonts and or graphics, which can be used to produce smooth text and lines in a chart. The CDL parameter, **AntiAlias = ON|ONDRAW|ONTEXT|OFF**, controls this behavior. **ON** will AnitAlias both text and lines. Starting with version 5.0, the default value is **ON**.

**Used in These Charts**
All

**Example:**

```plaintext
AntiAlias = ON;
```

**Attributes**

**mode**

**AutoscalePad**

**AutoscalePad = pad;**

**AutoscalePad** specifies a hint to the Netcharts logic that is used automatically determine an axis range. When no AxisScale parameters are specified for an axis, the axis automatically chooses a range based on the data values that are being plotted. **AutoscalePad** can be used to control these automatically chosen values.

**AutoscalePad** is specified as a percentage of the data range. For example, if a data series has a minimum value of 100 and a maximum value of 200, NetCharts will autoscale the axis to range from 100 to 200. If in addition **AutoscalePad=10** is specified, NetCharts will autoscale the axis to start at 10% below 100 and end at 10% above 200. If a minimum of maximum value is the data series is 0, that value will not be modified by **AutoscalePad**.
**Common CDL Attributes**

---

**Used in These Charts**
Bar, Box, Bubble, Combo, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**
```plaintext
AutoscalePad = 10;
```

**AxesSizes**

```plaintext
AxesSizes = (bottomsize, topsize, leftsize, rightsize);
```

**AxesSizes** specifies the amount of space that will be allocated to draw top, bottom, left and right axes. **AxesSizes** is typically used in conjunction with PlotArea to further control the layout of the PlotArea of a chart.

- **bottomsize** – amount of vertical space in chart allocated to the bottom axes. If bottomsize is a number between 0 and 1, it is interpreted as a percentage of the total plot area height. If bottomsize >= 1 it is interpreted as an absolute size in pixels.

- **topsize** – amount of vertical space in chart allocated to the top axes. If topsize is a number between 0 and 1 it is interpreted as a percentage of the total plot area height. If topsize >= 1 it is interpreted as an absolute size in pixels.

- **leftsize** – amount of horizontal space in chart allocated to the left axes. If leftsize is a number between 0 and 1, it is interpreted as a percentage of the total plot area width. If leftsize >= 1 it is interpreted as an absolute size in pixels.

- **rightsize** – amount of space in chart allocated to the right axes. If rightsize is a number between 0 and 1 it is interpreted as a percentage of the total plot area width. If rightsize >= 1 it is interpreted as an absolute size in pixels.

---

**Used in These Charts**
Bar, Box, Bubble, Combo, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**
```plaintext
AxesSizes = (1,1,1,1);
```

**Attributes**

- **bottomsize**
- **topsize**
- **leftsize**
- **rightsize**

**AxisThickness**

```plaintext
AxisThickness = thickness;
```

---

© Visual Mining, Inc 2002-2011
www.visualmining.com
The `AxisThickness` parameter defines the axis thickness. If thickness is greater than 1, then all axes will be displayed in 3D with the given thickness. This 3D look will only be active when one or more 3D grids have been specified. The default value is 0.

**Used in These Charts**
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**
```
AxisThickness = 15;
```

**Attributes**
`Thickness`

---

**AxisTitle**

The `AxisTitle` parameter specifies the label attributes for the axis title, which centered along the given axis, just above the grid. When the `Header` parameter, whether because of the use of a legend, or for some other reason, creates a title that seems visually unbalanced, you may find this parameter produces a more pleasing chart title.

<table>
<thead>
<tr>
<th>interiorAlignment</th>
<th>Specifies the alignment to use in text strings that contain multiple lines.</th>
</tr>
</thead>
<tbody>
<tr>
<td>exteriorAlignment</td>
<td>Specifies the alignment for the entire Title object.</td>
</tr>
</tbody>
</table>

The legal values for `interiorAlignment` and `exteriorAlignment` are `LEFT`, `RIGHT`, or `CENTER`.

**Used in These Charts**
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**
```
TopAxisTitle = ("Ceres Prototype Project Schedule\n ", black, "Helvetica", 12, LEFT);
```

**Attributes**
`Label`    `Color`    `FontName`    `FontSize`    `Angle`    `interiorAlignment`
AxisTitleActiveLabel

TopAxisTitleActiveLabel[N] = (“Label”, “URL”, “Target”), ...;
BottomAxisTitleActiveLabel[N] = (“Label”, “URL”, “Target”), ...;
LeftAxisTitleActiveLabel[N] = (“Label”, “URL”, “Target”), ...;
RightAxisTitleActiveLabel[N] = (“Label”, “URL”, “Target”), ...;

The AxisTitleActiveLabel parameter specified a custom active label to be associated with the axis title. That is, these labels will be displayed whenever the mouse “dwells” over the axis title.

Used in These Charts
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

Example:
AxisTitleActiveLabel = (“Label”, “URL”, “Target”);

Attributes
Label Target URL

AxisTitleBox

TopAxisTitleBox[N] = (Color, BorderType, BorderWidth, “ImageURL”, ImageFormat, BorderColor);
BottomAxisTitleBox[N] = (Color, BorderType, BorderWidth, “ImageURL”, ImageFormat, BorderColor);
LeftAxisTitleBox[N] = (Color, BorderType, BorderWidth, “ImageURL”, ImageFormat, BorderColor);
RightAxisTitleBox[N] = (Color, BorderType, BorderWidth, “ImageURL”, ImageFormat, BorderColor);

The AxisTitleBox parameter specifies the region attributes for the axis title centered along the axis. The image-related attributes need not be used unless you want to use an image texture on the box.

Used in These Charts
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

Example:
TopAxisTitleBox = (lightgray, SHADOW, 3,,gray);

Attributes
Color BorderType BorderWidth ImageURL
ImageFormat BorderColor
AxesGaps

AxesGaps = value, value, ...;

The AxesGaps parameter is used to specify the percentage of available space that should be allocated for gaps between axes. This parameter is used in conjunction with AxesLayout. The values specified in AxesLayout and AxesGaps are summed and each axis and gap is drawn in a space relative to its contribution to that sum. For example, LeftAxisLayout=20,40,20 and LeftAxisGaps=10,10 would assign 20% of the axis space to Axis1, 10% to the gap between Axis1 and Axis2, 40% to Axis2, 10% to the gap between Axis2 and Axis3 and 20% to Axis3.

Used in These Charts
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

Example:
AxesGaps = 2;

Attributes
Value

AxesLayout

AxesLayout = value1, value2, value3, ...;

AxesLayout is used to specify the percentage of available space that each axis will occupy.

Used in These Charts
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

Example:
AxesLayout = 85, 15;

Attributes
Value

AxesLayoutDirection

AxesLayoutDirection = mode, ...;
The `AxesLayoutDirection` parameter is used to specify the direction of the new axis, OUTWARD or NORMAL.

**Used in These Charts**
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**
```
AxesLayoutDirection = NORMAL;
```

**Values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NORMAL</td>
<td>Draws a 2D bar</td>
</tr>
<tr>
<td>OUTWARD</td>
<td></td>
</tr>
</tbody>
</table>

**Attributes**

- `Mode`

**Background**

```
Background = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);
```

The `Background` parameter, which is universal to Visual Mining charting programs, provides a visual background for the chart. It can have a border with a width of your choice, or have an image arranged in a variety of ways. We recommend that you use `Background` to make the chart blend in with your HTML page.

**Used in These Charts**
All

**Example:**
```
Background = (darkPink, NONE, 0, "/classes/netcharts/demo/flock.gif", SQUARE, SNIP, SQUARE, SNIP, white);
```

**Attributes**

- `BorderColor`
- `BorderType`
- `BorderWidth`
- `Color`
- `ImageFormat`
- `ImageURL`
- `TRCornerStyle`
- `BRCornerStyle`
- `BLCornerStyle`
- `CornerColor`
The drawing style for each of the four corners of the region. Styles are specified in a clockwise fashion starting in the upper left - Top Left, Top Right, Bottom Right, and Bottom Left. Legal values are SQUARE, SNIP and ROUND. The default is SQUARE.

**BackgroundActiveLabel**

BackgroundActiveLabel = ("Label", "URL", "Target");

BackgroundActiveLabel defines a single active label content and destination for the background of a chart.

**Used in These Charts**

All

**Example:**

BackgroundActiveLabel = ("Overview chart, click for details", "detail.html", "_framelm");

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
</table>

**BackgroundFillPattern**

BackgroundFillPattern = (type, color1, color2, imageURL), ...;

The BackgroundFillPattern parameter provides a visual pattern fill for the background of a chart.

<table>
<thead>
<tr>
<th>Type</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>no pattern, do default fill, if any</td>
</tr>
<tr>
<td><strong>Built-In Patterns</strong></td>
<td></td>
</tr>
<tr>
<td>FSLASH</td>
<td>front slash type</td>
</tr>
<tr>
<td>BS LASH</td>
<td>back slash type</td>
</tr>
<tr>
<td>DGRID</td>
<td>diagonal grid lines, (front and back slash lines)</td>
</tr>
<tr>
<td>HORIZONTAL</td>
<td>horizontal lines</td>
</tr>
<tr>
<td>VERTICAL</td>
<td>vertical lines</td>
</tr>
<tr>
<td>GRID</td>
<td>grid lines, (horizontal and vertical lines)</td>
</tr>
<tr>
<td><strong>Gradients</strong></td>
<td></td>
</tr>
<tr>
<td>GRADIENTVERTICAL</td>
<td>bottom to top style gradient</td>
</tr>
<tr>
<td>GRADIENTHORIZONTAL</td>
<td>left to right style gradient</td>
</tr>
<tr>
<td>GRADIENTFDIAG</td>
<td>top right to bottom left style gradient</td>
</tr>
<tr>
<td>GRADIENTBDIAG</td>
<td>top left to bottom right style gradient</td>
</tr>
<tr>
<td>GRADIENTRADIAL</td>
<td>radial style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERHORIZONTAL</td>
<td>center out horizontal style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERVERTICAL</td>
<td>center out vertical style gradient</td>
</tr>
</tbody>
</table>
Common CDL Attributes

<table>
<thead>
<tr>
<th>Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMAGE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>color1</th>
</tr>
</thead>
<tbody>
<tr>
<td>This color is used in the following ways:</td>
</tr>
<tr>
<td>- Foreground color for patterns</td>
</tr>
<tr>
<td>- Starting color for gradients</td>
</tr>
<tr>
<td>- Ignored in images</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>color 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>This color is used in the following ways:</td>
</tr>
<tr>
<td>- Background color for patterns</td>
</tr>
<tr>
<td>- Stopping color for gradients</td>
</tr>
<tr>
<td>- Ignored in images</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>imageURL</th>
</tr>
</thead>
<tbody>
<tr>
<td>The URL to an image to use as the fill</td>
</tr>
</tbody>
</table>

Used in These Charts
All

Example:
BackgroundFillPattern = (GRADIENTVERTICAL, blue, white);

Attributes
Type | Color1 | Color2 | ImageFormat | ImageURL
--- | --- | --- | --- | ---

Bar3DDepth

Bar3DDepth = Number;

Bar3DDepth defines the depth of the bars in a bar chart.

Used in These Charts
Bar, Combo, Pareto, Stock

Example:
Bar3DDepth = 10;

Attributes
Number

Number
Apparent depth of a bar in a bar chart, in pixels.

**Example:**

\[
\text{Bar3DDepth} = 5;
\]

**Values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Draws a 2D bar</td>
</tr>
<tr>
<td>1 or greater</td>
<td>Whole number depth in pixels</td>
</tr>
</tbody>
</table>

**Default**

10

**BarActiveLabels**

\[
\text{BarActiveLabels} = ("Label", "URL", "Target"), \ldots;
\]

Specifies sets of active labels attached to bars in a bar chart. Each grouped set in parentheses, or “tuple,” has a corresponding set within a `DataSet` parameter.

**Used in These Charts**

Bar, Combo, Pareto, Stock

**Example:**

\[
\text{BarActiveLabels} = ("OUTLINE", "Barchart8Mon.html", "InfoFrame"),
("OUTLINE", "Barchart8Tue.html", InfoFrame),
("No Log Info", "noinfo.html", InfoFrame);
\]

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
</table>

**BarAnimationStyle**

\[
\text{BarAnimationStyle} = \text{GROW} | \text{FADE} | \text{NONE}
\]

Specifies how the bars initially appear in a chart. This parameter is only valid in SVG or SVGWeb output formats.

**Attributes**

Style

**Style**

*Style* refers to the manner in which bars are first rendered in a bar chart.
Example:

```
BarAnimationStyle = GROW;
```

Values

GROW  The bars grow from the origin of the axis to their actual values.

FADE  The bars fade in.

NONE  The bars are immediately visible.

Default

NONE

BarBorder

```
BarBorder[N] = (LineType, LineWidth, Color);
```

For graphs using bars, `BarBorder` specifies the line style to be used for the border of all bars. The default line color is black.

Used in These Charts

Bar, Combo, Pareto, Stock

Example:

```
BarBorder = (DASHED, 2, DarkGray);
```

Attributes

<table>
<thead>
<tr>
<th>LineType</th>
<th>LineWidth</th>
<th>Color</th>
</tr>
</thead>
</table>

LineType

```
LineType specifies the style of the border to be drawn on a chart’s bars.
```

Example:

```
BarBorder = (DASHED, 2, DarkGray);
```

Values

SOLID  Draws a solid line of LineWidth thickness.

DOTTED  Draws a dotted line of LineWidth thickness.

DASHED  Draws a dashed line of LineWidth thickness.

DOTDASH  Draws a dot-dashed line of LineWidth thickness.

Default

SOLID
**LineWidth**

`LineWidth` specifies the width in pixels of the border to be drawn on a chart’s bars.

**Example:**

```
BarBorder = (DASHED, 2, DarkGray);
```

**Values**

1 or greater Whole number width in pixels

**Default**

1

**BarColorTable[n]**

```
BarColorTable[1-50] = Color1, Color2, Color3, Color4, Color5, ...;
```

`BarColorTable[n]` defines a set of colors for dataset N that overrides all other color specifications for that set. The parameters used for specifying the color of bars in a chart are (in ascending order of precedence) `ColorTable`, `DataSets`, `BarSymbol`, `BarFillPattern` and `BarColorTable`. `BarColorTable` is used most frequently to color some specific bar.

For example

```
BarColorTable2 = ,,blue;
```

will change the third bar in the second series to blue, while all other bars in the chart continue to be colored by one of the other color related parameters.

The colors you can use are defined in the common `Color` attribute (Chapter 4).

**Used in These Charts**

Bar, Combo, Pareto, Stock

**Example:**

```
BarColorTable2 = ,,red;
```

**Attributes**

None

**BarColorTable[n]P[m]**

```
BarColorTable[1-50]P[1-50] = Color1, Color2, Color3, Color4, Color5, ...;
```

`BarColorTable[n]P[m]` is used only for grouped stacked bar charts (GraphType=GROUPSTACK). It defines a set of colors for dataset N that overrides all other color specifications for that set. The parameters used for specifying the color of bars in a chart are (in ascending order of precedence) `ColorTable`,...
DataSets, BarSymbol, BarFillPattern and BarColorTable[1-50]M[1-50]. BarColorTable is used most frequently to color some specific bar.

For example:

```
BarColorTable2P1 = blue,blue;
```

will change the color the first bar of the second data series blue at the first tic location and at the second tic location.

**BarCorners**

```
BarCorners = Number;
```

BarCorners specifies the corner style to be used for the 2D bars in a chart. The attributes correspond to specific corners of the bar beginning at the top left and advancing in clockwise sequence to the bottom left. The default corner style is SQUARE. Elements which can be selected for each corner are: SQUARE, SNIP and ROUND. SNIP trims the square corner from the bar obliquely. ROUND substitutes a smoothly rounded edge for the square corner. The amount to be snipped or rounded can be adjusted by adding an underscore modifier to the elements SNIP and ROUND followed by a value. A whole number value will set the number of pixels to be removed. A fractional value will set the percentage of the overall bar to be removed. To set the percentage, enter a fractional value between 0 and 1 and place it after the underscore (e.g. “SNIP_0.09” will cause 9% of the corner to be removed; “ROUND_0.50” will cause 50% of the corner to be rounded). When using a fractional value, enclose the tuple element in double-quotes to “escape” the decimal point.

**Used in These Charts**

Bar, Combo, Pareto, Stock

**Example:**

```
BarCorners = ("SNIP_0.09","SNIP_0.09",SQUARE,SQUARE);
```

**Attributes**

<table>
<thead>
<tr>
<th>TopLeft</th>
<th>TopRight</th>
<th>BottomRight</th>
<th>BottomLeft</th>
</tr>
</thead>
</table>

**TopLeft**

TopLeft specifies the style and size of the upper left corner to be drawn on a chart’s bars.

**Example:**

```
BarCorners = (SNIP,SQUARE,SQUARE,SQUARE);
```
Values

- SQUARE: Draws a square corner.
- SNIP: Cuts the corner off obliquely.
- ROUND: Draws a rounded corner.
- SNIP_val: Cuts the corner off obliquely based upon the number of pixels or percentage shown in val.
- ROUND_val: Draws a rounded corner based upon the number of pixels or percentage shown in val.

Default

SQUARE

TopRight

TopRight specifies the style and size of the upper right corner to be drawn on a chart’s bars.

Example:
```
BarCorners = (SQUARE,SNIP,SQUARE,SQUARE);
```

BottomRight

BottomRight specifies the style and size of the lower right corner to be drawn on a chart’s bars.

Example:
```
BarCorners = (SQUARE,SQUARE,SNIP,SQUARE);
```

BottomLeft

BottomLeft specifies the style and size of the lower left corner to be drawn on a chart’s bars.

Example:
```
BarCorners = (SQUARE,SQUARE,SNIP,SQUARE);
```

BarDropShadow

BarDropShadow = (color, offsetx, offsety, size);

BarDropShadow places a shadow on the background field of the bar chart. The color, orientation, and size of the shadow can be defined. The tuple element color sets the color of the shadow. That color value is interpolated to complete transparency as it reaches the end of the shadow’s blur area. Offsetx and offsety define the center point of the shadow; offsetx sets the x-axis offset from the chart’s center-point; offsety sets the y-axis offset. When an offset attribute is set to a whole number value, the position...
of the drop shadow is offset from the chart’s center point by the number of pixels set by that whole number. When an offset is set to a fractional value (between 0 and 1), the value is interpreted as a percentage of the width of the object casting the shadow (the set of bars). The center of the drop shadow is repositioned based upon the values or percentages set for offsetx and offsety. Offset attribute values may be positive or negative. Size sets the size of the blur area, the region beyond the actual drop shadow shape where the shadow is extended and blurred into transparency. The size of this blurred region is controlled by the size attribute. The blurred region becomes larger and more diffuse as the value of size is increased. When size is set to a whole number value, the size of the blurred area is defined in pixels. When size is set to a fractional value (between 0 and 1), the value is interpreted as a percentage of the width of the object casting the shadow. When using a fractional value, enclose the value in double-quotes to “escape” the decimal point.

**Used in These Charts**
Bar, Combo, Pareto

**Example:**
```
BarDropShadow = (color, offsetx, offsety, size);
```

**Attributes**
- Color
- Offsetx
- Offsety
- Size

**Color**

Color specifies the base color of the shadow to be drawn behind a chart’s bars.

**Example:**
```
BarDropShadow = (black, “-0.05”, “-0.05”, 55);
```

**Offsetx**

Offsetx specifies the x-coordinate offset from center.

**Example:**
```
BarDropShadow = (black, 25 -10, 25);
```

**Offsety**

Offsety specifies the y-coordinate offset from center.

**Example:**
```
BarDropShadow = (black, 25 -10, 25);
```
Size

Size specifies the width of the blurred area.

Example:

BarDropShadow = (black, "-.05", "-.05", 55);

BarFillPattern

BarFillPattern[N] = (type, color1, color2, imageURL), ...;

The BarFillPattern parameter provides a visual pattern fill for bars in a bar or combo chart.

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>no pattern, do default fill, if any</td>
</tr>
<tr>
<td><strong>Built-In Patterns</strong></td>
<td></td>
</tr>
<tr>
<td>FSLASH</td>
<td>front slash type</td>
</tr>
<tr>
<td>BSLASH</td>
<td>back slash type</td>
</tr>
<tr>
<td>DGRID</td>
<td>diagonal grid lines, (front and back slash lines)</td>
</tr>
<tr>
<td>HORIZONTAL</td>
<td>horizontal lines</td>
</tr>
<tr>
<td>VERTICAL</td>
<td>vertical lines</td>
</tr>
<tr>
<td>GRID</td>
<td>grid lines, (horizontal and vertical lines)</td>
</tr>
<tr>
<td><strong>Gradients</strong></td>
<td></td>
</tr>
<tr>
<td>GRADIENTVERTICAL</td>
<td>bottom to top style gradient</td>
</tr>
<tr>
<td>GRADIENTHORIZONTAL</td>
<td>left to right style gradient</td>
</tr>
<tr>
<td>GRADIENTFDIAG</td>
<td>top right to bottom left style gradient</td>
</tr>
<tr>
<td>GRADIENTBDIAG</td>
<td>top left to bottom right style gradient</td>
</tr>
<tr>
<td>GRADIENTRADIAL</td>
<td>radial style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERHORIZONTAL</td>
<td>center out horizontal style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERVERTICAL</td>
<td>center out vertical style gradient</td>
</tr>
<tr>
<td><strong>Images</strong></td>
<td></td>
</tr>
<tr>
<td>IMAGE</td>
<td>use an image specified in the optional imageURL element</td>
</tr>
</tbody>
</table>

**color1**

This color is used in the following ways:
- Foreground color for patterns
- Starting color for gradients
- Ignored in images

**color 2**

This color is used in the following ways:
- Background color for patterns
- Stopping color for gradients
- Ignored in images

**imageURL**

The URL to an image to use as the fill.
**Common CDL Attributes**

**Size**

**Used in These Charts**
Bar, Combo, Pareto

**Example:**

```
BarFillPattern = (GRADIENTVERTICAL, blue, white), ...;
```

**Attributes**

<table>
<thead>
<tr>
<th>Type</th>
<th>Color1</th>
<th>Color2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageFormat</td>
<td>ImageURL</td>
<td></td>
</tr>
</tbody>
</table>

**BarHighlights**

```
BarHighlights = (type, start, stop, top, right, bottom, left, width, height), ...;
```

The `BarHighlights` parameter provides a visual pattern fill in a bar or combo chart. It adds or overlays a fill pattern over one or more existing fill patterns to produce multiple effects. The width or height of the pattern can be modified. The gap between the sides of the bar and the fill pattern being applied can be modified. Gradient patterns can be set using the `type` attribute. Only gradient patterns may be used. A type value of NONE suppresses the highlights. The element `start` sets the beginning color of the gradient; the element `stop` sets the end color of the gradient. Color values are interpolated between the two. The elements `top`, `right`, `bottom`, and `left` specify the size of the gaps between the edge of the highlight and the associated edge of the bar. When the values for `top`, `right`, `bottom`, and `left` are specified as whole numbers, they set the distance between the edge of the highlight and the edge of the bar in pixels. When set to a fractional number between 0 and 1, they set the gap to a percentage of the width of the bar. When using a fractional value, enclose the tuple element in double-quotes to “escape” the decimal point.

When using a fractional value, `width` controls the width of the highlight; the element `height` controls the height. When values for `width` and `height` are specified using whole numbers, they set the distance in pixels. If `width` or `height` is set to -1, `BarHighlights` fills all of the space available after taking into account any gaps specified in previous attributes. If `width` or `height` is set to a fractional number between 0 and 1, `BarHighlights` sets the width or height of the highlight using the percentage of available width or height of the bar.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>no pattern, do default fill, if any</td>
</tr>
<tr>
<td>Gradients</td>
<td></td>
</tr>
<tr>
<td>GRADIENTVERTICAL</td>
<td>bottom to top style gradient</td>
</tr>
<tr>
<td>GRADIENTHORIZONTAL</td>
<td>left to right style gradient</td>
</tr>
<tr>
<td>GRADIENTFDIAG</td>
<td>top right to bottom left style gradient</td>
</tr>
<tr>
<td>GRADIENTBDIAG</td>
<td>top left to bottom right style gradient</td>
</tr>
<tr>
<td>GRADIENTRADIAL</td>
<td>radial style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERHORIZONTAL</td>
<td>center out horizontal style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERVERTICAL</td>
<td>center out vertical style gradient</td>
</tr>
</tbody>
</table>

**start**

This color is used in the following
Common CDL Attributes

<table>
<thead>
<tr>
<th>ways:</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Starting color for gradients</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>stop</th>
</tr>
</thead>
<tbody>
<tr>
<td>This color is used in the following ways:</td>
</tr>
<tr>
<td>- Stopping color for gradients</td>
</tr>
</tbody>
</table>

**Used in These Charts**
Bar, Combo, Pareto, Stock

**Example:**
```
BarHighlights = (GRADIENTRADIAL, yellow, white, 15, 15, 15, 15, -1, -1);
```

**Attributes**

<table>
<thead>
<tr>
<th>type</th>
<th>start</th>
<th>stop</th>
</tr>
</thead>
<tbody>
<tr>
<td>top</td>
<td>right</td>
<td>left</td>
</tr>
<tr>
<td>width</td>
<td>height</td>
<td></td>
</tr>
</tbody>
</table>

**BarRightFillPattern**

**Attributes**

| BarRightFillPattern = (type, color1, color2, imageURL), ...; |

The **BarRightFillPattern** parameter provides a visual pattern fill for the right surface of a bar in a 3D bar 3D combo or Pareto chart.

<table>
<thead>
<tr>
<th>Type</th>
<th>Built-In Patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>no pattern, do default fill, if any</td>
</tr>
<tr>
<td>FSLASH</td>
<td>front slash type</td>
</tr>
<tr>
<td>BSLASH</td>
<td>back slash type</td>
</tr>
<tr>
<td>DGRID</td>
<td>diagonal grid lines, (front and back slash lines)</td>
</tr>
<tr>
<td>HORIZONTAL</td>
<td>horizontal lines</td>
</tr>
<tr>
<td>VERTICAL</td>
<td>vertical lines</td>
</tr>
<tr>
<td>GRID</td>
<td>grid lines, (horizontal and vertical lines)</td>
</tr>
</tbody>
</table>

**Gradients**

| GRADIENTVERTICAL | bottom to top style gradient |
| GRADIENTHORIZONTAL | left to right style gradient |
| GRADIENTFDIAG | top right to bottom left style gradient |
| GRADIENTBDIAG | top left to bottom right style gradient |
| GRADIENTRADIAL | radial style gradient |
| GRADIENTCENTERHORIZONTAL | center out horizontal style gradient |
| GRADIENTCENTERVERTICAL | center out vertical style gradient |

**Images**

| IMAGE | use an image specified in the optional imageUrl element |

<table>
<thead>
<tr>
<th>color1</th>
</tr>
</thead>
<tbody>
<tr>
<td>This color is used in the following ways:</td>
</tr>
</tbody>
</table>
Common CDL Attributes

<table>
<thead>
<tr>
<th>color 2</th>
<th>This color is used in the following ways:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- Background color for patterns</td>
</tr>
<tr>
<td></td>
<td>- Stopping color for gradients</td>
</tr>
<tr>
<td></td>
<td>- Ignored in images</td>
</tr>
</tbody>
</table>

```
Example:
BarRightFillPattern = (GRADIENTVERTICAL, blue, white), ...;
```

Used in These Charts
Bar, Combo, Pareto, Stock

Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Color1</th>
<th>Color2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageFormat</td>
<td>ImageURL</td>
<td></td>
</tr>
</tbody>
</table>

BarRightFillPattern[n]P[m]

```
BarRightFillPattern[1-50]P[1-50] = (type, color1, color2, imageURL), ...;
```

The `BarRightFillPattern[n]P[m]` parameter provides a visual pattern fill for the right surface of a bar in a 3D stacked bar 3D combo chart or Pareto chart. Used for bar series in grouped stacked barcharts (GraphType=GROUPSTACK). In grouped stacked barcharts, `BarRightFillPattern[n]P[m]` defines the right fill pattern applied to a 3D stack at a single tic location. See also: `DataSet[n]P[m]`.

<table>
<thead>
<tr>
<th>Type</th>
<th>NONE</th>
<th>no pattern, do default fill, if any</th>
</tr>
</thead>
<tbody>
<tr>
<td>Built-In Patterns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FSLASH</td>
<td>front slash type</td>
<td></td>
</tr>
<tr>
<td>BSLASH</td>
<td>back slash type</td>
<td></td>
</tr>
<tr>
<td>DGRID</td>
<td>diagonal grid lines, (front and back slash lines)</td>
<td></td>
</tr>
<tr>
<td>HORIZONTAL</td>
<td>horizontal lines</td>
<td></td>
</tr>
<tr>
<td>VERTICAL</td>
<td>vertical lines</td>
<td></td>
</tr>
<tr>
<td>GRID</td>
<td>grid lines, (horizontal and vertical lines)</td>
<td></td>
</tr>
</tbody>
</table>

Gradients

<table>
<thead>
<tr>
<th>Type</th>
<th>GRADIENTVERTICAL</th>
<th>bottom to top style gradient</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GRADIENTHORIZONTAL</td>
<td>left to right style gradient</td>
</tr>
<tr>
<td></td>
<td>GRADIENTFDIAG</td>
<td>top right to bottom left style gradient</td>
</tr>
<tr>
<td></td>
<td>GRADIENTBDIAG</td>
<td>top left to bottom right style gradient</td>
</tr>
<tr>
<td></td>
<td>GRADIENTRADIAL</td>
<td>radial style gradient</td>
</tr>
</tbody>
</table>
Common CDL Attributes

<table>
<thead>
<tr>
<th>GRADIENTCENTERHORIZONTAL</th>
<th>center out horizontal style gradient</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRADIENTCENTERVERTICAL</td>
<td>center out vertical style gradient</td>
</tr>
</tbody>
</table>

**Images**

| IMAGE | use an image specified in the optional imageURL element |

<table>
<thead>
<tr>
<th>color1</th>
<th>This color is used in the following ways:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- Foreground color for patterns</td>
</tr>
<tr>
<td></td>
<td>- Starting color for gradients</td>
</tr>
<tr>
<td></td>
<td>- Ignored in images</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>color 2</th>
<th>This color is used in the following ways:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- Background color for patterns</td>
</tr>
<tr>
<td></td>
<td>- Stopping color for gradients</td>
</tr>
<tr>
<td></td>
<td>- Ignored in images</td>
</tr>
</tbody>
</table>

| imageURL | The URL to an image to use as the fill |

**Used in These Charts**

StackedBar

**Example:**

```
BarTopRightFillPattern1P2 = (GRADIENTVERTICAL, blue, white), ...;
```

**Attributes**

<table>
<thead>
<tr>
<th>Type</th>
<th>Color1</th>
<th>Color2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageFormat</td>
<td></td>
<td></td>
</tr>
<tr>
<td>imageURL</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**BarSpotlights**

```
BarSpotlights = (start, stop, center, centeroffsetx, centeroffsety, focusoffsetx, focusoffsety, radius), ...;
```

Adds or overlays a color fill over one or more existing fill patterns to produce multiple layered color effects. The spotlight “illuminates” the bars of the bar or combo chart. The center of the spotlight and its focus can be adjusted independently by adjusting offsets from the bar chart center point. The elements `centeroffsetx` and `centeroffsety` set the x and y-coordinates of the center of the spotlight as an offset of the chart center point. When set to whole numbers, `centeroffsetx` and `centeroffsety` specify the number of pixels to offset from the chart center point. When set to fractional values (between 0 and 1), they are interpreted as percentages of the width of the bars. Percentage values are written using a decimal point (e.g. “0.05” and “0.50” represent 5% and 50%, respectively). When using a fractional value, enclose the tuple element in double-quotes to “escape” the decimal point. The element `focusoffsetx` is the offset from the chart center which defines the x-coordinate of the focus point of the spotlight. The element `focusoffsety` is the offset from the chart center which defines the y-coordinate of the focus point of the spotlight. When set to whole numbers, `focusoffsetx` and `focusoffsety` specify the offset from the center in pixels. When set to fractional values (between 0 and 1), they are interpreted as percentages of the width of the bars. The element `radius` sets the size of the spotlight, from its center to its edge. When set to a whole number, it sets the size of...
the \textit{radius} in pixels; when set to a fractional value, it sets the radial diameter of the spotlight based upon that percentage of the minimum height and width of the bars on the chart.

\textbf{Used in These Charts}

Bar, Combo, Pareto, Stock

\textbf{Example:}

\begin{verbatim}
BarSpotlights = (purple_40, blue_155, RIGHT, 50, -50, 100, 150, 250);
\end{verbatim}

\textbf{Attributes}

\begin{verbatim}
start       stop       center
centeroffsetx     centeroffsety     focusoffsetx
focusoffsety       radius
\end{verbatim}

\textbf{Start}

\textit{Start} specifies the first of the two colors which will be interpolated to produce a gradient spotlight.

\textbf{Example:}

\begin{verbatim}
BarSpotlights = (purple_40, blue_155, LEFT, 20, 10, 120, -120, 250);
\end{verbatim}

\textbf{Stop}

\textit{Stop} specifies the second of two colors which will be interpolated to produce a gradient spotlight.

\textbf{Example:}

\begin{verbatim}
BarBorder = (DASHED, 2, DarkGray);
\end{verbatim}

\textbf{Center}

\textit{Center} specifies the position around of the center of the chart where the spotlight center will be placed.

\textbf{Example:}

\begin{verbatim}
BarSpotlights = (purple_40, blue_155, LEFT, 20, 10, 120, -120, 250);
\end{verbatim}
Values

RIGHT Offsets the center point of the spotlight to the right of the center point of the chart.
LEFT Offsets the center point to the left.
TOP Offsets the center point to the top.
BOTTOM Offsets the center point to the bottom.
CENTER Uses the chart center point for the spotlight center point.
TOPRIGHT Offsets the center point of the spotlight to the top right.
TOPLEFT Offsets the center point of the spotlight to the top left.
BOTTOMRIGHT Offsets the center point of the spotlight to the bottom right.
BOTTOMLEFT Offsets the center point of the spotlight to the bottom left.

Default
CENTER

Centeroffsetx

Centeroffsetx specifies the x-coordinate offset for the spotlight center.

Example:

BarSpotlights = (purple_40,blue_155,LEFT,20,10,120,-120,250);

Centeroffsety

Centeroffsety specifies the y-coordinate offset for the spotlight center.

Example:

BarSpotlights = (purple_40,blue_155,LEFT,20,10,120,-120,250);

Focusoffsetx

Focusoffsetx specifies the x-coordinate offset for the center of the spotlight’s focus.

Example:

BarSpotlights = (purple_40,blue_155,LEFT,20,10,120,-120,250);

Focusoffsety

Focusoffsety specifies the y-coordinate offset for the center of the spotlight’s focus.

Example:

BarSpotlights = (purple_40,blue_155,LEFT,20,10,120,-120,250);
**Radius**

Radius specifies the length of the radius of the spotlight from the center of the spotlight.

*Example:*

```
BarSpotlights = (purple_40,blue_155,LEFT,20,10,120,-120,250);
```

**BarStyle**

BarStyle[N] = (mode, color, font name, width); Remember to change these VALUES

Defines the label value to be displayed on each group of bars. This parameter overrides the BottomLabels parameter (for VERTICAL bars) and the LeftLabels parameter (for HORIZONTAL bars).

*Used in These Charts*

Bar, Combo, Pareto

*Example:*

```
BarValueLabel = ("ON", black, "Helvetica", 18);
```

**Attributes**

<table>
<thead>
<tr>
<th>Mode</th>
<th>Color</th>
<th>Font Name</th>
<th>Width</th>
</tr>
</thead>
</table>

**BarSymbol**

BarSymbol = (BarSymbolType, BarSymbolColor)...;

Defines the style and color of the bars in a barset. Values specified in this parameter override values set in the DataSets parameter.

*Used in These Charts*

Bar, Combo

*Example:*

```
BarSymbol = (CYLINDER,blue),(RECTANGLE,red);
```

**Attributes**

<table>
<thead>
<tr>
<th>BarSymbolType</th>
<th>BarSymbolColor</th>
</tr>
</thead>
</table>

**BarSymbolType**

The visualization style for the bars in a barset. Legal values are BAR, TRIANGLEBAR, DIAMONDBAR, CYLINDER, PIEHORIZONTAL and PIEVERTICAL. The default value is BAR.
BarTopFillPattern

BarTopFillPattern[N] = (type, color1, color2, imageURL), ...;

The BarTopFillPattern parameter provides a visual pattern fill for the top surface of a bar in a 3D bar 3D combo or Pareto chart.

<table>
<thead>
<tr>
<th>Type</th>
<th>NONE</th>
<th>no pattern, do default fill, if any</th>
</tr>
</thead>
<tbody>
<tr>
<td>Built-In Patterns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FSLASH</td>
<td>front slash type</td>
<td></td>
</tr>
<tr>
<td>BSLASH</td>
<td>back slash type</td>
<td></td>
</tr>
<tr>
<td>DGRID</td>
<td>diagonal grid lines, (front and back slash lines)</td>
<td></td>
</tr>
<tr>
<td>HORIZONTAL</td>
<td>horizontal lines</td>
<td></td>
</tr>
<tr>
<td>VERTICAL</td>
<td>vertical lines</td>
<td></td>
</tr>
<tr>
<td>GRID</td>
<td>grid lines, (horizontal and vertical lines)</td>
<td></td>
</tr>
<tr>
<td>Gradients</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRADIENTVERTICAL</td>
<td>bottom to top style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTHORIZONTAL</td>
<td>left to right style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTFDIAG</td>
<td>top right to bottom left style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTBDIAG</td>
<td>top left to bottom right style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTRADIAL</td>
<td>radial style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTCENTERHORIZONTAL</td>
<td>center out horizontal style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTCENTERVERTICAL</td>
<td>center out vertical style gradient</td>
<td></td>
</tr>
<tr>
<td>Images</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IMAGE</td>
<td>use an image specified in the optional imageURL element</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>color1</th>
<th>This color is used in the following ways:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- Foreground color for patterns</td>
</tr>
<tr>
<td></td>
<td>- Starting color for gradients</td>
</tr>
<tr>
<td></td>
<td>- Ignored in images</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>color2</th>
<th>This color is used in the following ways:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- Background color for patterns</td>
</tr>
<tr>
<td></td>
<td>- Stopping color for gradients</td>
</tr>
<tr>
<td></td>
<td>- Ignored in images</td>
</tr>
</tbody>
</table>

| imageURL      | The URL to an image to use as the fill   |
**Example:**

\[
\text{BarTopFillPattern} = (\text{GRADIENTVERTICAL, blue, white}), \ldots;
\]

### Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Color1</th>
<th>Color2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageFormat</td>
<td>ImageURL</td>
<td></td>
</tr>
</tbody>
</table>

### BarTopFillPattern[n]P[m]

\[
\text{BarTopFillPattern}[1-50]P[1-50] = (\text{type, color1, color2, imageURL}), \ldots;
\]

The `BarTopFillPattern[n]P[m]` parameter provides a visual pattern fill for the top surface of a bar in a 3D stacked bar 3D combo chart or Pareto chart. Used for bar series in grouped stacked barcharts (GraphType=GROUPSTACK). In grouped stacked barcharts, `BarTopFillPattern[n]P[m]` defines the top fill pattern applied to a 3D stack at a single tic location. See also: `DataSet[n]P[m]`.

<table>
<thead>
<tr>
<th>Type</th>
<th>No pattern, do default fill, if any</th>
</tr>
</thead>
<tbody>
<tr>
<td>Built-In Patterns</td>
<td></td>
</tr>
<tr>
<td>FSLASH</td>
<td>front slash type</td>
</tr>
<tr>
<td>BSLASH</td>
<td>back slash type</td>
</tr>
<tr>
<td>DGRID</td>
<td>diagonal grid lines, (front and back slash lines)</td>
</tr>
<tr>
<td>HORIZONTAL</td>
<td>horizontal lines</td>
</tr>
<tr>
<td>VERTICAL</td>
<td>vertical lines</td>
</tr>
<tr>
<td>GRID</td>
<td>grid lines, (horizontal and vertical lines)</td>
</tr>
<tr>
<td>Gradients</td>
<td></td>
</tr>
<tr>
<td>GRADIENTVERTICAL</td>
<td>bottom to top style gradient</td>
</tr>
<tr>
<td>GRADIENTHORIZONTAL</td>
<td>left to right style gradient</td>
</tr>
<tr>
<td>GRADIENTFDIAG</td>
<td>top right to bottom left style gradient</td>
</tr>
<tr>
<td>GRADIENTBDIAG</td>
<td>top left to bottom right style gradient</td>
</tr>
<tr>
<td>GRADIENTRADIAL</td>
<td>radial style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERHORIZONTAL</td>
<td>center out horizontal style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERVERTICAL</td>
<td>center out vertical style gradient</td>
</tr>
<tr>
<td>Images</td>
<td></td>
</tr>
<tr>
<td>IMAGE</td>
<td>use an image specified in the optional imageURL element</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>color1</th>
<th>This color is used in the following ways:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Foreground color for patterns</td>
</tr>
<tr>
<td></td>
<td>Starting color for gradients</td>
</tr>
<tr>
<td></td>
<td>Ignored in images</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>color 2</th>
<th>This color is used in the following ways:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Background color for patterns</td>
</tr>
<tr>
<td></td>
<td>Stopping color for gradients</td>
</tr>
<tr>
<td></td>
<td>Ignored in images</td>
</tr>
</tbody>
</table>

| imageURL   | The URL to an image to use as the fill   |
Common CDL Attributes

BarSymbolType

Used in These Charts
StackedBar

Example:

BarTopTopFillPattern1P2 = (GRADIENTVERTICAL,blue,white), ...;

Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Color1</th>
<th>Color2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageFormat</td>
<td>ImageURL</td>
<td></td>
</tr>
</tbody>
</table>

BarValueLabel

BarValueLabel[N] = (mode, color, font name, width);

Defines the label value to be displayed on each group of bars. This parameter overrides the BottomLabels parameter (for VERTICAL bars) and the LeftLabels parameter (for HORIZONTAL bars).

Used in These Charts
Bar, Combo, Pareto

Example:

BarValueLabel = ("ON", black, "Helvetica", 18);

Attributes

| Mode   | Color | Font Name | Width |

BarValueLabelBox

BarValueLabelBox = (color, mode, depth);

Defines the label box to be displayed with each group of bars.

Used in These Charts
Bar, Combo, Pareto

Example:

BarValueLabelBox = (grey, RAISED, 3);

Attributes

<table>
<thead>
<tr>
<th>Color</th>
<th>Mode</th>
<th>Depth</th>
</tr>
</thead>
</table>
BarValueLabelStyle

\[ \text{BarValueLabelStyle} = \text{labelposition}_1, \text{labelposition}_2, \ldots \text{labelposition}_N; \]

Defines where the BarValueLabel text will display for each dataset.

*Used in These Charts*
Bar, Combo, Pareto

*Example:*

\[ \text{BarValueLabelStyle} = \text{MIDDLE}, \text{MIDDLE}, \text{MIDDLE}; \]

*Attributes*
Label Position

BarWidth

\[ \text{BarWidth} = \text{Percent}_1, \text{Percent}_2, \ldots \text{Percent}_N; \]

Defines the relative width of the bars in a barset as a percentage of available space. Legal values are 1-100. A smaller number results in more space between bars. Width\(_N\) defines the Width for DataSet\(_N\).

*Used in These Charts*
Bar, Combo, Histogram, Pareto, Stock

*Example:*

\[ \text{BarWidth} = 60,30; \quad \text{<!- Barset1 bars fills 60% of the space available -->} \]

*Attributes*

BestFit

\[ \text{BestFit} = \text{ON}|\text{OFF} \]

BestFit when turned ON attempts to auto fit each pie and its labels into a square.

*Used in These Charts*
MultiPie

*Example:*

\[ \text{BestFit} = \text{ON} \]
**BottomActiveLabels**

\[
\text{BottomActiveLabels} = ("Label", "URL", "Target"), \ldots;
\]

The bottom axis labels become active labels when this parameter is used. Each set in parenthesis has a corresponding set within a `DataSet` parameter.

**Used in These Charts**

Bar, Box, Bubble, Combo, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**

```
BottomActiveLabels = ("", ".//cgibin/LA01.cgi", "frame1"),
("OUTLINE", ".//cgibin/LA02.cgi", "frame1"),
("", ".//cgibin/LA03.cgi", "frame1");
```

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
</table>

**BottomAxisTitleActiveLabel**

\[
\text{BottomAxisTitleActiveLabel} = ("Label", "URL", "Target");
\]

The `BottomAxisTitleActiveLabel` parameter defines a single active label destination for the `BottomAxisTitle` parameter. Using this element also requires use of the `ActiveClicks` parameter if you wish to navigate from this interaction.

**Used in These Charts**

All

**Example:**

```
BottomAxisTitleActiveLabel = ("Destination", "demo.html", "frame1");
```

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
</table>

**BottomAxisTitleBox**

\[
\text{BottomAxisTitleBox} = (\text{Color}, \text{BorderType}, \text{BorderWidth}, "\text{ImageURL}", \text{ImageFormat}, \text{BorderColor});
\]

The `BottomAxisTitleBox` parameter specifies the region attributes for the axis title centered along the axis. The image-related attributes need not be used unless you want to use an image texture on the box.
**Common CDL Attributes**

**BarSymbolType**

**Used in These Charts**
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**
```plaintext
BottomAxisTitleBox = ("lightgray, SHADOW, 3,,,gray);
```

**Attributes**

<table>
<thead>
<tr>
<th>Color</th>
<th>BorderType</th>
<th>BorderWidth</th>
<th>ImageURL</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageFormat</td>
<td>BorderColor</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**BottomColor**

`BottomColor = Color;`

`BottomColor` controls the color of the bottom axis and the tic marks, but not the tic mark labels. The default axis color is black. If the NULL color is specified, the axis color is not changed by this parameter.

**Example:**
```plaintext
BottomColor = xB5D5F0;
```

**Attributes**

Color

**BottomDrawMinorTics**

`BottomDrawMinorTics = ON|OFF;`

`BottomDrawMinorTics` controls whether or not bottom tics are drawn. The default value is ON.

**Example:**
```plaintext
BottomDrawMinorTics = OFF;
```

**Attributes**

(Switch)

**BottomFormat**

`BottomFormat = (FormatType, "FormatExpr", "TimeBase", "TimeUnit");`

The `BottomFormat` parameter defines the format of an axis’ numeric tic labels.

**Used in These Charts**
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y
Example:

```plaintext
BottomFormat = (INTEGER);
BottomFormat = (FLOAT, "$%9.2f", ,);
BottomFormat = (DATE, "%M/%y", "1/1/00 12:00", "1M");
BottomFormat = (INTEGER, "$%dK");
```

**Attributes**

<table>
<thead>
<tr>
<th>FormatType</th>
<th>FormatExpr</th>
<th>TimeBase</th>
<th>TimeUnit</th>
</tr>
</thead>
</table>

**BottomLabels**

BottomLabels = "Label1", "Label2", ...;

The BottomLabels parameter specifies a list of custom tic mark labels that will be used instead of the numeric labels automatically generated by the axis. The BottomLabels will be evenly placed along the axis, overriding any tic placement specified by the StepValue attribute.

In a Bar, Combo, Pareto, or Stock Chart, the BarLabels parameter overrides the BottomLabels (for vertical bars) parameters.

**Used in These Charts**
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**

```plaintext
```

**Attributes**

Label

**BottomMargins**

BottomMargins = (LeftSideMargin, RightSideMargin);

The BottomMargins parameter specifies the gap, in pixels, at the beginning and end of the bottom axis. Most often used to prevent clipping of data points at the extreme ends of the scale.

**Example:**

```plaintext
BottomMargins = (20, 20);
```

**BottomScale**

BottomScale = (MinValue, MaxValue, StepValue);
The `BottomScale` parameter specifies the minimum and maximum data values which will be displayed along the bottom axis. If the `BottomScale` parameter is not defined, or the `MinValue` and `MaxValue` parameters are the same or one of them is not defined, then the tick mark locations will be automatically determined based on the actual data values being displayed. That is, the axis will be "autoscaled" using the current data values to determine "reasonable" values for `MinValue`, `MaxValue`, and `StepValue`. If values are supplied for any of `MinValue`, `MaxValue`, or `StepSize`, those values will be used as part of the autoscaling.

In Bar charts, the `BottomScale` parameter automatically determined by the number of bars being displayed.

**Used in These Charts**
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Time, X-Y

**Example:**
```plaintext
BottomScale = ("1 Apr 96", "1 Jun 96", "14d");
```

**Attributes**

<table>
<thead>
<tr>
<th>MinValue</th>
<th>MaxValue</th>
<th>StepValue</th>
</tr>
</thead>
</table>

**MinValue**

MinValue sets the absolute lower visible limit for the bottom axis scale.

**Example:**
```plaintext
BottomScale = ("1 Apr 96", "1 Jun 96", "14d");
```

**Values**
Any real number, date, or time less than `MaxValue`

**Default**
None

**MaxValue**

ScrollMax sets the absolute upper visible limit for the bottom axis scale.

**Example:**
```plaintext
BottomScale = ("1 Apr 96", "1 Jun 96", "14d");
```

**Values**
Any real number, date, or time greater than `MinValue`

**Default**
None
**StepValue**

StepValue is optional, and may be used to specify a given step between tic marks along the bottom axis, starting with the MinValue. If StepValue is not an even multiple of the difference between the MinValue and MaxValue, then no tic mark will be displayed at the MaxValue. If StepValue is not defined, tic marks will be placed at "reasonable" locations along the axis, depending on the range of values being displayed.

**Example:**

```
BottomScale = ("1 Apr 96", "1 Jun 96", "14d");
```

**Values**

Any real number, date, or time between MinValue and MaxValue

**Default**

1

**BottomScroll**

```
BottomScroll = (ScrollMin, ScrollMax);
```

The BottomScroll parameter specifies a range of values through which the bottom axis can be scrolled. When the ScrollMin and ScrollMax attributes are defined for the axis, the axis will be displayed as a slider bar, using the axis color defined, with a white background. The relative size of the slider represents the percentage of the entire range currently being displayed. That is, it graphically depicts the size of the current axis range (MinValue and MaxValue attributes) relative to the scrollable region (ScrollMin and ScrollMax attributes). See the BottomScale parameter for MinValue and MaxValue definitions.

**Example:**

```
BottomScroll = (0, 98);
```

**Attributes**

ScrollMin ScrollMax

**ScrollMin**

ScrollMin sets the lower visible limit for a scrollbar defined with BottomScroll

**Example:**

```
BottomScroll = (0, 98);
```

**Values**

<MinValue
Default
None

ScrollMax

ScrollMax sets the upper visible limit for a scrollbar defined with BottomScroll

Example:
    BottomScroll = (0, 98);

Values
> MaxValue

Default
None

BottomTicLength

BottomTicLength = Number;

The BottomTicLength parameter defines the size of axis tic marks which are displayed along the bottom axis of a chart. The parameter will reset the automatically generated tic length. The value defines the number of pixels to use for the length of the tic mark. By default, the number of pixels used is the width of the character zero (0) as found in the font applied to the label. Setting the BottomTicLength to the value -1 will cause the default size to be used.

Attributes
Number

Number

Apparent length of a bottom axis tic mark in a chart, in pixels.

Used in These Charts
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

Example:
    BottomTicLength = 10;

Values
0 No effect (zero length tics are not drawn).
1 or greater Whole number length in pixels
**Default**
-1

**BottomTics**

```
BottomTics = ("Mode", Color, "FontName", FontSize, Angle, interiorAlignment, backgroundColor, rotationPoint);
```

The `BottomTics` parameter specifies the label attributes for the tic marks displayed along the bottom axis. The tic labels are generated automatically based on the other parameter settings, and are displayed using the given label attributes in the `BottomTics` parameter. If any attribute is not defined, any previous value of that attribute will be used.

<table>
<thead>
<tr>
<th>interiorAlignment</th>
<th>Specifies the alignment to use in text strings that contain multiple lines.</th>
</tr>
</thead>
<tbody>
<tr>
<td>backgroundColor</td>
<td>Background color of the tic label area</td>
</tr>
<tr>
<td>rotationPoint</td>
<td>For rotated axis labels, anchor point for the rotation</td>
</tr>
</tbody>
</table>

The legal values for `interiorAlignment` are `LEFT`, `RIGHT`, or `CENTER`.

The legal values for `rotationPoint` are `LEFT`, `RIGHT`.

**Used in These Charts**
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**
```
BottomTics = ("ON", black, "Helvetica", 10, LEFT, null, RIGHT);
```

**Attributes**

<table>
<thead>
<tr>
<th>Mode</th>
<th>Color</th>
<th>FontName</th>
<th>FontSize</th>
<th>Angle</th>
</tr>
</thead>
<tbody>
<tr>
<td>InteriorAlignment</td>
<td>BackgroundColor</td>
<td>RotationPoint</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Mode**

Mode determines whether or not the tic labels are shown on that axis.

**Example:**
```
BottomTics = ("ON", black, "Helvetica", 10, LEFT, null, RIGHT);
```

**Values**

- **ON** Show tic labels for this axis
- **OFF** Don’t show tic labels on this axis

**Default**
ON
**BottomTitle**

BottomTitle = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment, exteriorAlignment);

BottomTitle describes an optional title, or label, that sits on the bottom of a chart, and uses standard attributes for string text, text color, font, font size, and label rotation. As with, BottomTitle is universally available in Visual Mining chart applications.

<table>
<thead>
<tr>
<th>interiorAlignment</th>
<th>Specifies the alignment to use in text strings that contain multiple lines.</th>
</tr>
</thead>
<tbody>
<tr>
<td>exteriorAlignment</td>
<td>Specifies the alignment for the entire Title object.</td>
</tr>
</tbody>
</table>

The legal values for interiorAlignment and exteriorAlignment are LEFT, RIGHT, or CENTER.

**Used in These Charts**

All, but most useful in Pie, Diagram, and Dial, which do not have axes.

**Example:**

```plaintext
BottomTitle = ("Financial Status", royalblue, Helvetica, 14, 0);
```

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>Color</th>
<th>FontName</th>
<th>FontSize</th>
<th>Angle</th>
<th>interiorAlignment</th>
</tr>
</thead>
</table>

**BottomTitleBox**

BottomTitleBox = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor);

The BottomTitleBox specifies a background region just for the BottomTitle parameter.

**Used in These Charts**

All

**Attributes**

<table>
<thead>
<tr>
<th>BorderColor</th>
<th>BorderType</th>
<th>BorderWidth</th>
<th>Color</th>
<th>ImageFormat</th>
<th>ImageURL</th>
</tr>
</thead>
</table>

**BoxActiveLabels**

BoxActiveLabels = ("Label","URL","Target"), ...;

The BoxActiveLabels parameter specified a custom active label to be associated with the BoxLabels on a box chart. That is, these labels will be displayed whenever the mouse “dwells” over the optional name of a box data series.
**Used in These Charts**

**Box**

**Example:**

```plaintext
BoxActiveLabels = ("Set1", "URL", "Target");
```

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>Target</th>
<th>URL</th>
</tr>
</thead>
</table>

**BoxFence**

```plaintext
BoxFence = Mode;
```

**BoxFence** specifies whether or not to draw fences at the ends of the Inter Quartile Range (the box).

**Used in These Charts**

**Box Chart**

**Example:**

```plaintext
BoxFence = ON;
BoxFence = OFF;
```

**Attributes**

<table>
<thead>
<tr>
<th>Type</th>
</tr>
</thead>
</table>

**Type**

**Type** refers to the fences at the ends of the box.

**Example:**

```plaintext
BoxFence = ON;
```

**Values**

<table>
<thead>
<tr>
<th>ON</th>
<th>Draw fences at the ends of the Inter Quartile Range, (the box).</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFF</td>
<td>Do not draw fences.</td>
</tr>
</tbody>
</table>

**Default**

<table>
<thead>
<tr>
<th>ON</th>
</tr>
</thead>
</table>
**BoxFillPattern**

BoxFillPattern[N] = (type, color1, color2, imageURL), ...;

The **BoxFillPattern** parameter provides a visual pattern fill for a box displayed in a chart.

<table>
<thead>
<tr>
<th>Type</th>
<th>NONE</th>
<th>no pattern, do default fill, if any</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Built-In Patterns</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FSLASH</td>
<td>front slash type</td>
<td></td>
</tr>
<tr>
<td>BSLASH</td>
<td>back slash type</td>
<td></td>
</tr>
<tr>
<td>DGRID</td>
<td>diagonal grid lines, (front and back slash lines)</td>
<td></td>
</tr>
<tr>
<td>HORIZONTAL</td>
<td>horizontal lines</td>
<td></td>
</tr>
<tr>
<td>VERTICAL</td>
<td>vertical lines</td>
<td></td>
</tr>
<tr>
<td>GRID</td>
<td>grid lines, (horizontal and vertical lines)</td>
<td></td>
</tr>
<tr>
<td><strong>Gradients</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRADIENTVERTICAL</td>
<td>bottom to top style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTHORIZONTAL</td>
<td>left to right style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTFDIAG</td>
<td>top right to bottom left style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTBDIAG</td>
<td>top left to bottom right style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTRADIAL</td>
<td>radial style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTCENTERHORIZONTAL</td>
<td>center out horizontal style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTCENTERVERTICAL</td>
<td>center out vertical style gradient</td>
<td></td>
</tr>
<tr>
<td><strong>Images</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IMAGE</td>
<td>use an image specified in the optional imageURL element</td>
<td></td>
</tr>
</tbody>
</table>

- **color1**
  - This color is used in the following ways:
  - Foreground color for patterns
  - Starting color for gradients
  - Ignored in images

- **color 2**
  - This color is used in the following ways:
  - Background color for patterns
  - Stopping color for gradients
  - Ignored in images

- **imageURL**
  - The URL to an image to use as the fill

**Used in These Charts**

Box

**Example:**

BoxFillPattern = (GRADIENTVERTICAL,blue,white);
**BoxHeight**

BoxHeight\[N\] = \textit{Height};

BoxHeight describes the height (in pixels) for every box displayed.

**Used in These Charts**

Box Chart

**Example:**

\[
\text{Boxheight} = 5;
\]

**Attributes**

**Height**

Height determines box height in whole pixels.

**Example:**

\[
\text{Boxheight} = 5;
\]

**Values**

- 0: default height will be selected based on the size of the display and the number of data sets being displayed

  The default height will also be used if the specified height is taller than the amount of space that is physically available to a given box.

**Default**

Calculated value

**BoxLabels**

BoxLabels = \texttt{"Label1","Label2", ...};
The BoxLabels parameter specifies optional label to be associated with the data series on a box chart. If the GraphLayout of the BoxChart is VERTICAL then BoxLabels will be displayed on the Bottom Axis. If the GraphLayout of the BoxChart is HORIZONTAL then BoxLabels will be displayed on the Left Axis.

**Used in These Charts**
Box

**Example:**
BoxLabels = “Set1”, “Set2”;

---

**BoxLimitLines**

BoxLimitLines = (limit1-1, limit1-2,...limit1-N),...(limitM-1, limitM-2,... limitM-N);

BoxLimitLines allows the user to specify the values for the limit lines for individual data series.

**Used in These Charts**
Box Chart

**Example:**
BoxLimitLines = (2,4,6),(3,6,9),...;

**Attributes**
Limit

---

**BoxLimitLineStyle**

BoxLimitLineStyle = (type1,width1,color1),...(typeN,widthN,colorN);

BoxLimitLineStyles defines the look of the limit lines for each data series. All limit lines for a single data series must have the same style.

The legal values for type are SOLID, DASHED,DOTTED and DOTDASH.

**Used in These Charts**
Box Chart

**Example:**
BoxLimitLineStyle = (SOLID,1,red),(SOLID,1,green);
Attributes
Type   Width  Color

**BoxSymbolWidth**

BoxSymbolWidth = *percentage*;

BoxSymbolWidth controls the width of the box relative to the width of the fences. It is specified as a percent. A value of 100 would cause the box width and fence width to be equal.

**Used in These Charts**
Box Chart

**Example:**

    BoxSymbolWidth = 95;

Attributes
Width

**Default**
95

**BoxWidth**

BoxWidth = *percentage*;

BoxWidth controls the width of the box and fences relative to the space allotted to the data series. It is specified as a percent. If a dataset is allocated 100 pixels of space in the chart, setting the BoxWidth to 50 would cause the box and fences to use half of that space for their width.

**Used in These Charts**
Box Chart

**Example:**

    BoxWidth = 75;

Attributes
Width
**Default**

75

**BubbleAnimationStyle**

BubbleAnimationStyle = GROW | FADE | NONE

Specifies how the bubbles initially appear in a bubble chart. This parameter is only valid in SVG or SVGWeb output formats.

**Attributes**

- **Style**

**Style**

Style refers to the manner in which bubbles are first rendered in a bubble chart.

**Example:**

BubbleAnimationStyle = GROW;

**Values**

- **GROW**
  The bubbles grow from a diameter of zero to their actual diameters.

- **FADE**
  The bubbles fade in.

- **NONE**
  The bubbles are immediately visible.

**Default**

NONE

**BubbleAxis**

BubbleAxis[N] = (XAxis1, YAxis1), (XAxis2, YAxis2), ...;

BubbleAxis indicates one or more axes for the bubble graph, which must correspond to matching data sets.

**Used in These Charts**

Bubble

**Example:**

BubbleAxis = (BOTTOM, LEFT), (BOTTOM, RIGHT);

**Attributes**

XAxis Yaxis
BubbleColorTable

BubbleColorTable[1-50] = Color1, Color2, Color3, Color4, Color5, ...

BubbleColorTable defines a set of colors for dataset N that overrides all other color specifications for that set. The parameters used for specifying the color of bubbles in a chart are (in ascending order of precedence) ColorTable, DataSets, BubbleSymbol, BubbleFillPattern and BubbleColorTable. BarColorTable is used most frequently to color some specific bar.

For example

BubbleColorTable2 = ,,blue;

will change the third bubble in the second series to blue, while all other bubbles in the chart continue to be colored by one of the other color related parameters.

The colors you can use are defined in the common Color attribute (Chapter 4).

Used in These Charts

Bubble

Example:

BubbleColorTable2 = ,,red;

Attributes

None

BubbleFillPattern

BubbleFillPattern[N] = (type, color1, color2, imageURL), ...;

The BubbleFillPattern parameter provides a visual pattern fill inside the bubble area of a chart.

<table>
<thead>
<tr>
<th>Type</th>
<th>NONE</th>
<th>no pattern, do default fill, if any</th>
</tr>
</thead>
<tbody>
<tr>
<td>Built-In Patterns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FSLASH</td>
<td>front slash type</td>
<td></td>
</tr>
<tr>
<td>BSLASH</td>
<td>back slash type</td>
<td></td>
</tr>
<tr>
<td>DGRID</td>
<td>diagonal grid lines, (front and back slash lines)</td>
<td></td>
</tr>
<tr>
<td>HORIZONTAL</td>
<td>horizontal lines</td>
<td></td>
</tr>
<tr>
<td>VERTICAL</td>
<td>vertical lines</td>
<td></td>
</tr>
<tr>
<td>GRID</td>
<td>grid lines, (horizontal and vertical lines)</td>
<td></td>
</tr>
<tr>
<td>Gradients</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRADIENTVERTICAL</td>
<td>bottom to top style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTHORIZONTAL</td>
<td>left to right style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTFDIAG</td>
<td>top right to bottom left style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTBDIAG</td>
<td>top left to bottom right style gradient</td>
<td></td>
</tr>
</tbody>
</table>
### Common CDL Attributes

<table>
<thead>
<tr>
<th>Style</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRADIENTRADIAL</td>
<td>radial style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERHORIZONTAL</td>
<td>center out horizontal style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERVERTICAL</td>
<td>center out vertical style gradient</td>
</tr>
</tbody>
</table>

### Images

<table>
<thead>
<tr>
<th>IMAGES</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMAGE</td>
<td>use an image specified in the optional imageURL element</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>color1</strong></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This color is used in the following ways:</td>
</tr>
<tr>
<td></td>
<td>- Foreground color for patterns</td>
</tr>
<tr>
<td></td>
<td>- Starting color for gradients</td>
</tr>
<tr>
<td></td>
<td>- Ignored in images</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>color 2</strong></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This color is used in the following ways:</td>
</tr>
<tr>
<td></td>
<td>- Background color for patterns</td>
</tr>
<tr>
<td></td>
<td>- Stopping color for gradients</td>
</tr>
<tr>
<td></td>
<td>- Ignored in images</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>imageURL</strong></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The URL to an image to use as the fill</td>
</tr>
</tbody>
</table>

---

### Used in These Charts

**Bubble**

**Example:**

```
BubbleFillPattern = (GRADIENTFDIAG,blue,white);
```

**Attributes**

<table>
<thead>
<tr>
<th>Type</th>
<th>Color1</th>
<th>Color2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageFormat</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ImageURL</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### BubbleScale

```
BubbleScale[N] = (MinValue, MaxValue, AREA|DIAMETER, PointColor), ...;
```

**BubbleScale** controls the bubble scale to be displayed for the named bubble sets.

**Used in These Charts**

**Bubble**

**Example:**

```
BubbleScale   = (10,100,DIAMETER,white),(100,500,AREA,green);
```
## Attributes

<table>
<thead>
<tr>
<th>MinValue</th>
<th>MaxValue</th>
<th>AREA or DIAMETER</th>
<th>PointColor</th>
</tr>
</thead>
</table>

### MinValue

MinValue specifies the minimum Z value that will result in the display of a bubble. Data points with Z values smaller than minValue will not be drawn.

**Example:**

```
BubbleScale = (10,100,DIAMETER,white), (100,500,AREA,green);
```

**Values**

Any real number $\geq 0$.

**Default**

None

### MaxValue

MaxValue is the maximum value of z in the BubbleSet, which is displayed with a symbol the size of MaxSize. The sizes of bubbles with other Z values will be drawn proportionally.

**Example:**

```
BubbleScale = (10,100,DIAMETER,white), (100,500,AREA,green);
```

**Values**

Any real number $\geq 0$.

**Default**

None

### Area or Diameter

This switch specifies how the relative sizes of bubbles with a Z value $< \text{maxValue}$ are determined. For AREA, the area of the bubbles is proportional to the Z value defined. For DIAMETER, the diameter of the bubbles is proportional to the Z value defined.

**Example:**

```
BubbleScale = (10,100,DIAMETER,white), (100,500,AREA,green);
```

**Values**

One of AREA or DIAMETER
**Default**
None

**PointColor**

If `PointColor` is specified, a dot is drawn in the center of the bubble in the color specified. Otherwise, this attribute behaves as does the common `Color` attribute.

**Example:**

```
BubbleScale   = (10,100,DIAMETER,white),(100,500,AREA,green);
```

**Values**

See the common `Color` attribute in Chapter 4 for details.

**Default**
blue

**BubbleSets**

```
BubbleSets[N] = ("Name1", Color1), ("Name2", Color2),...;
```

BubbleSets defines the data sets for a bubble graph; these data sets are (X,Y,Z) that can be rendered as a series of bubbles— or other symbols, such as squares or diamond—connected by optional lines. The bubbles are graphed along defined X and Y axes, with a third coordinate, Z, that determines the relative size of the bubble.

**Used in These Charts**

Bubble

**Example:**

```
BubbleSets = ("Server #1",x0572c6),("Server #2",xE32F41);
```

**Attributes**

Name    Color

**Name**

Name is the value assigned to a data point from the BubbleSets parameter, and represents an (X,Y,Z) numeric vector.

**Example:**

```
BubbleSets = ("Server #1",x0572c6),("Server #2",xE32F41);
```

**Values**

Any string value
**Default**
None

**BubbleSet[n]**

BubbleSetn = (x,y,z), (x,y,z), ...;

Defines a vector of (x,y,z) values for the named bubble set defined by the BubbleSets parameter. The z value defines the relative size of the bubble within the BubbleSet. If NULL is substituted for any part of the vector, the bubble will not be drawn. A value of z equal to zero results in having no bubble drawn, as well.

**Used in These Charts**
Bubble

**Example:**

BubbleSet1 = (1,27,10), (5,50,20), (10,100,30), (23,125,40), (56,170,50),
           (65,220,60), (68,280,70);
BubbleSet2 = (3,27,100), (5,40,200), (8,125,300), (26,137,400), (75,260,500);

**Attributes**
None

**BubbleSymbol**

BubbleSymbol[n] = (SymType, MaxSize, Style, BorderColor, BorderWidth,
SymbolColor, ShadowWidth), ...;

BubbleSymbol controls the display of the bubbles or symbols for the named bubble sets. You should specify as many groups, or “tuples,” as there are bubble sets.

**Used in These Charts**
Bubble

**Example:**

BubbleSymbol = (CIRCLE,80,FILLED,null,null),(SQUARE,60,OUTLINED,xe3e3e3,1);

**Attributes**

<table>
<thead>
<tr>
<th>SymType</th>
<th>MaxSize</th>
<th>Style</th>
<th>BorderColor</th>
<th>BorderWidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>ShadowThickness</td>
<td>SymbolColor</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SymbolColor**
SymbolColor specifies the foreground color of the LineSymbol.

**ShadowThickness**

ShadowThickness specifies the size of the shadow behind a BubbleSymbol. Any number other than 0 causes NetCharts to choose a shadow size based on the size of the symbol. Specifying a value of 0 suppresses the shadow.

**SymType**

SymType indicates the type of symbol displayed for these bubbles.

**Example:**

```plaintext
BubbleSymbol = (CIRCLE, 80, FILLED, null, null), (SQUARE, 60, OUTLINED, xe3e3e3, 1);
```

**Values**

- **CIRCLE** Displays circles
- **SQUARE** Displays squares
- **DIAMOND** Displays diamonds
- **CROSS** Displays crosses
- **TARGET** Displays targets (bulls-eye)
- **TRIANGLE_DOWN** Displays downward pointing triangles
- **TRIANGLE_UP** Displays upward pointing triangles

**Default**

CIRCLE

**MaxSize**

MaxSize indicates the maximum size of the bubble symbol, in pixels.

**Example:**

```plaintext
BubbleSymbol = (CIRCLE, 80, FILLED, null, null), (SQUARE, 60, OUTLINED, xe3e3e3, 1);
```

**Values**

Any whole pixel value.

**Default**

None

**Style**

Style controls how the bubble symbol should be drawn.

**Example:**

```plaintext
BubbleSymbol = (CIRCLE, 80, FILLED, null, null), (SQUARE, 60, OUTLINED, xe3e3e3, 1);
```
### Values

- **FILLED**: symbol is filled with the bubble set color
- **OUTLINED**: only the outline is drawn, using the bubble set color
- **BOTH**: symbol is filled with the bubble set color and the outline is drawn using the(BorderColor).

### Default

OUTLINED

### BuildAnimationEnabled

\[
\text{BuildAnimationEnabled} = \text{ON} \mid \text{OFF};
\]

BuildAnimationEnabled enables or disables all animation on charts that are delivered in SVG format.

### Used in These Charts

ALL

**Example:**

\[
\text{BuildAnimationEnabled} = \text{OFF};
\]

### Attributes

**Mode**

### CellTextAutoColorThreshold

\[
\text{CellTextAutoColorThreshold} = \text{range}
\]

Color distance threshold (between the grid foreground and grid background colors) that determines whether a foreground text color swap is necessary for visibility.

### Used in These Charts

Heat Map

**Example:**

\[
\text{CellTextAutoColorThreshold} = 20;
\]

### Attributes

**Range**

### Values

0-100 Percent
**CenterRadius**

\[ \text{CenterRadius}[N] = (radius); \]

The \text{CenterRadius} parameter defines the diameter of the circle at the origin of the chart.

**Used in These Charts**

Radar

**Example:**

\[
\text{CenterRadius} = 15; \]

**Attributes**

\text{Radius}

---

**ChartElementSpacing**

\text{ChartElementSpacing} = \text{spacing};

\text{ChartElementSpacing} defines the space between the elements of a chart. Chart elements include Titles, Legends, Axes and the PlotArea.

\text{spacing} – the size in pixels of the horizontal and vertical space between chart elements.

**Used in These Charts**

Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**

\[
\text{ChartElementSpacing} = 3; \]

**Attributes**

\text{Spacing}

---

**Charts**

\text{Charts} = (\text{Name}1, \text{Type}1, \text{Width}1, \text{Height}1), (\text{Name}2, \text{Type}2, \text{Width}2, \text{Height}2),...;
The Charts parameter is the main support for the display of multiple charts within a Multi-chart. It defines the name and type for each chart to be displayed. For each chart defined in this parameter, the corresponding ChartURL and ChartScript parameters will be processed to define the chart itself.

If a chart name or type of BREAK is specified, then the subsequent charts will be displayed on a new row or column. In this way, multiple rows or columns with a different number of charts may be displayed. If the optional Width and Height attributes are defined, they are used to determine the percentage of space that should be allocated to each chart.

IMPORTANT NOTE: Because the Multi-chart allows the definition of multiple ChartScripts, the standard NFParamScript parameter is not supported. Consequently, all parameters must be defined using the standard <param> tag within the applet parameter, as shown in the examples below.

**Used in These Charts**
Multi-chart

**Example:**

```
<PARAM NAME=Charts VALUE="
("Piechart1",PIECHART),
("Piechart2",PIECHART),
("Piechart3",PIECHART);'>

<PARAM NAME=Charts VALUE="
("Bar1",BARCHART),
("Bar2",BARCHART),
(BREAK),
("Bar3",BARCHART),
("Bar4",BARCHART)>}

<PARAM NAME=Layout VALUE="{ROWS}">

<PARAM NAME=Charts VALUE="
("Sales",PIECHART),
("Expense",BARCHART),
("Bonus",XYCHART),
(BREAK),
("Growth",COMBOCHART),
(BREAK),
("Budget",XYCHART),
("Salary",PIECHART);'

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Width</th>
<th>Height</th>
</tr>
</thead>
</table>

**Type**

Type specifies the type of chart shown as one element of the multi-chart.

**Example:**

```
<PARAM NAME=Charts VALUE="
("Piechart1",PIECHART),
("Linechart",LINECHART),
("Piechart2",PIECHART);'
```
Values

BARCHART
BOXCHART
COMBOCHART
DIAGRAM
DIALCHART
LINECHART
PARETOCHART
PIECHART
STOCKCHART
STRIPCHART
TIMECHART
XYCHART

Default
None. You must use one of the values above.

Width, Height

The Width and Height attributes are optional, and specify the width and height of the component charts. The Width and Height attributes are interpreted as "relative" sizes, depending on the total number of charts in each row or column. If no Width or Height attributes are specified, then a default value of 1 is used. It is easiest to specify the width and height as percentages in the range 1 to 100, although that is not required.

Only the first chart in each row need have the width and height specified, since those values will be used as the defaults for the subsequent charts. To get uniform widths in each row, you can specify 1 for the width of each chart.

Example:

```xml
<PARAM NAME=Layout VALUE="(ROWS)">  <!-- charts laid out in rows -->
<PARAM NAME=Charts VALUE="
  ("Sales", PIECHART, 1, 25),  <!-- gets 25% of chart height -->
  ("Expense", BARCHART),
  ("Bonus", XYCHART),
  (BREAK),
  ("Growth", COMBOCHART, 1, 50),  <!-- gets 50% of chart height -->
  (BREAK),
  ("Budget", XYCHART, 1, 25),  <!-- gets 25% of chart height -->
  ("Salary", PIECHART);

`
```

Values

1 to 100     Relative sizes, in percentages

Default

1       Makes all widths and heights uniform
ChartHeight

ChartHeight = height;

The ChartHeight parameter allows a chart writer to specify the height of a chart.

*Used in These Charts*

All

*Example:*

    ChartHeight = 200;

*Attributes*

Height

ChartName

ChartName = name;

The ChartName parameter allows a chart writer to specify the name of a chart.

*Used in These Charts*

All

*Example:*

    ChartName = "Chart XYZ";

*Attributes*

Name

ChartScript[n]

ChartScript[1-20] = "parameter definition script";

The ChartScript parameter takes the place of the standard NFParamScript parameter for Multi-charts. It is similar, though, in that it allows a developer to specify any number of parameters for the chart.

In the Multi-chart, each ChartScript parameter defines the parameters for the chart types defined in the Charts parameter. Accordingly, care should be taken to ensure that the appropriate types of parameters are defined for a given chart type. Use the format in the example, below.

*Used in These Charts*

Multi-chart

*Example:*

    <PARAM NAME=Charts VALUE='("Bar1",BARCHART), ("Bar2",BARCHART), (BREAK),...}
Attributes

Insert the appropriate chart parameters within ChartScript.

ChartType

ChartType = type;

The ChartType parameter allows a chart writer to specify the type of chart to be used.

Used in These Charts

All

Example:

ChartType = barchart;

Attributes

Type

ChartURL[n]

ChartURL[1-20] = "URL";

ChartURL specifies the location of the chart definition file to be used to define the corresponding chart. The URL may use either the HTTP or FILE protocol, depending on the browser environment in which the applet is running. In all cases, if a relative URL is given, then the document base of the applet will be used to locate the relative file.

The ChartURL parameter may be used in addition to or instead of the ChartScript parameter. If both are specified, then the ChartURL parameter is processed first, allowing developers to specify default
parameters using the URL file, which can be overridden by the parameters defined in the ChartScript parameter.

**Used in These Charts**

Multi-chart

**Example:**

```plaintext
ChartURL1 = "chart1.cdl";
```

**Attributes**

- **URL**

---

**ChartWidth**

```plaintext
ChartWidth = width;
```

The **ChartWidth** parameter allows a chart writer to specify the width of a chart.

**Used in These Charts**

All

**Example:**

```plaintext
ChartWidth = 250;
```

**Attributes**

- **Width**

---

**Color**

```plaintext
TopColor[N] = Color;
BottomColor[N] = Color;
LeftColor[N] = Color;
RightColor[N] = Color;
```

The **Color** parameter controls the color of the given axis and the tic marks, but not the tic mark labels. The default axis color is black. If the **NULL** color is specified, the axis color is not changed by this parameter.

**Example:**

```plaintext
BottomColor = xB5D5F0;
```

**Attributes**

- **Color**
**ColorTable**

\[ \text{ColorTable} = \text{Color1}, \text{Color2}, \text{Color3}, \text{Color4}, \text{Color5}, \ldots; \]

`ColorTable` is a powerful way to control the appearance of charts and impose both uniformity and color harmony upon them. `ColorTable` supercedes whatever system color table is in use. The colors in a color table will repeat in sequence whenever the number of data sets exceeds the number of colors defined in the `ColorTable` parameter.

The colors you can use are defined in the common `Color` attribute (Chapter 4).

**Used in These Charts**

All

**Example:**

\[ \text{ColorTable} = \text{xB5D5F0}, \text{xBEA9AD}, \text{xDACE98}, \text{xEBF0F3}, \text{xAABAC5}, \text{xBFC1A0}, \text{xF2E0D4}; \]

**Attributes**

None

---

**CumulativeLineSetName**

\[ \text{CumulativeLinesetName} = \text{name}; \]

Name assigned to the cumulative line set. Used in the legend if the Legend CDL parameter does not define a label for the cumulative line.

**Used in These Charts**

Pareto

**Example:**

\[ \text{CumulativeLineSetName} = \text{“Cumulative Percentage Line”} \]

---

**CumulativeLineStyle**

\[ \text{CumulativeLineStyle} = (\text{Type}, \text{LineWidth}, \text{Color}, \text{FillColor}, \text{LineType}); \]

This parameter specifies the line style to be displayed for the cumulative percentage line.
**Common CDL Attributes**

**FillColor**

*Used in These Charts*

Pareto

*Example:*

`CumulativeLineStyle = (SOLID, 3, blue, blue, NORMAL);`

*Attributes*

<table>
<thead>
<tr>
<th>Color</th>
<th>FillColor</th>
<th>LineType</th>
<th>LineWidth</th>
<th>Type</th>
</tr>
</thead>
</table>

**FillColor**

If this attribute is not **NULL**, then the area under the cumulative line will be filled with the given color.

*Example:*

`CumulativeLineStyle = (SOLID, 3, red, pink, NORMAL);`

<-- red line with pink fill -->

*Values*

- NULL: Also, value left unspecified: No color fills the area under the line.
- Any legal color: Area under the line is filled. See Chapter 4 for the Color attribute.

*Default*

None

**Type**

The style of line to draw

*Values*

- NONE
- SOLID
- DOTTED
- DASHED
- DOTDASH

*Default*

SOLID

**LineType**

The type of line to use to connect the points in the line set.

*Values*

- NORMAL
- FIT
- BOTH
Default
NORMAL

CumulativeLineSymbol

```
CumulativeLineSymbol = (Type, Size, Style, BorderColor, BorderWidth, ImageURL, SymbolColor, ShadowWidth);
```

CumulativeLineSymbol specifies the symbols to be displayed for the cumulative line.

*Used in These Charts*

Pareto

*Example:*

```
CumulativeLineSymbol = (CIRCLE, 6, BOTH, white, 1, grey, 0);
```

**Attributes**

<table>
<thead>
<tr>
<th>BorderColor</th>
<th>BorderWidth</th>
<th>ImageURL</th>
<th>ShadowWidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>Style</td>
<td>SymbolColor</td>
<td>Type</td>
</tr>
</tbody>
</table>

**Size**

Size specifies the size of the symbol in pixels. This attribute is ignored for IMAGE symbols.

*Example:*

```
CumulativeLineSymbol = (CIRCLE, 6, BOTH, white, 1);
```

**Values**

Any integer value in pixels

**Default**

None

**SymbolColor**

SymbolColor specifies forground color of the CumulativeLineSymbol.

**ShadowThickness**

ShadowThickness specifies for size of the shadow behind a CumulativeLineSymbol. Any number other than 0 causes NetCharts to choose a shadow size based on the size of the symbol. Specifying a value of 0 suppresses the shadow.
**Style**

*Style* specifies how the *LineSymbol* should be drawn, including *FILLED*, *OUTLINED*, or *BOTH*. If *FILLED* is specified, the symbol is filled with the line set color. If *OUTLINED* is specified, only the outline is drawn, using the line set color. If *BOTH* is specified, then the symbol is filled with the line set color and the outline is drawn using the *borderColor*.

*Example:*

```plaintext
CumulativeLineSymbol = (CIRCLE, 6, BOTH, white, red);
```

**Values**

- **FILLED**: Symbol is filled with the cumulative line color.
- **OUTLINED**: Only the outline is drawn, using the cumulative line color.
- **BOTH**: Symbol is filled with the cumulative line color and the outline is drawn using the *BorderColor*.

**Default**

None

**Type**

Type specifies the type of symbol to be displayed on the line set.

*Example:*

```plaintext
CumulativeLineSymbol = (SQUARE, 6, BOTH, cyan, 1);
```

**Values**

- **NONE**: No symbol is displayed.
- **CIRCLE**: Displays circles
- **SQUARE**: Displays squares
- **DIAMOND**: Displays diamonds
- **CROSS**: Displays crosses
- **TARGET**: Displays targets (bulls-eye)
- **TRIANGLEDOWN**: Displays downward pointing triangles
- **TRIANGLEUP**: Displays upward pointing triangles
- **IMAGE**: If specified, the *ImageURL* attribute is required and will be used to load a GIF image for the symbol.

**Default**

None

**CumulativeLineValueLabel**

```plaintext
CumulativeLineValueLabel = (mode, color, font name, width);
```

Defines the label value to be displayed for each point in the cumulative line.
Common CDL Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Used in These Charts</th>
<th>Example</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pareto</td>
<td></td>
<td>Mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Color</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Font Name</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Width</td>
</tr>
</tbody>
</table>

### CumulativeLineValueLabelBox

CumulativeLineValueLabelBox = (color, mode, depth);

Defines the line label box to be displayed with each point in the cumulative line.

#### Used in These Charts

Pareto

#### Example:

```
CumulativeLineValueLabelBox = (grey, RAISED, 3);
```

#### Attributes

<table>
<thead>
<tr>
<th>Color</th>
<th>Mode</th>
<th>Depth</th>
</tr>
</thead>
</table>

### CumulativeLineValueLabelStyle

CumulativeLineValueLabelStyle = labelposition;

Defines where the line value label text will display for each point in the cumulative line.

#### Used in These Charts

Pareto

#### Example:

```
CumulativeLineValueLabelStyle = TOP;
```
### Values

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOPLEFT</td>
<td>Displayed at the top left point</td>
</tr>
<tr>
<td>TOP</td>
<td>Displayed at the top of the point</td>
</tr>
<tr>
<td>TOPRIGHT</td>
<td>Displayed at the top right point</td>
</tr>
<tr>
<td>LEFT</td>
<td>Displayed at the left of the point</td>
</tr>
<tr>
<td>CENTER</td>
<td>Displayed at the center point</td>
</tr>
<tr>
<td>RIGHT</td>
<td>Displayed at the right of the point</td>
</tr>
<tr>
<td>BOTTOMLEFT</td>
<td>Displayed at the bottom left point</td>
</tr>
<tr>
<td>BOTTOM</td>
<td>Displayed at the bottom of the point</td>
</tr>
<tr>
<td>BOTTOMRIGHT</td>
<td>Displayed at the bottom right point</td>
</tr>
</tbody>
</table>

**Default**

**TOP**

**Attributes**

**Label Position**

### DataAxis

\[
\text{DataAxis}[N] = (XAxis_1, YAxis_1), (XAxis_2, YAxis_2), \ldots;
\]

DataAxis defines the X and Y axes for associated data sets in charts using bars.

**Used in These Charts**

Bar, Box, Combo, Pareto, Stock, Strip, Time

**Example:**

```
DataAxis = (BOTTOM, LEFT), (BOTTOM, RIGHT);
```

**Attributes**

**XAxis**  **YAxis**

### DataLegend

DataLegend = ON|OFF;

DataLegend enables the display of a Data Legend on a Bar, Line or Combo chart. A Data Legend will present the chart data values in a table that is combined with the Bottom or Left Axis tic labels. The position of the DataLegend depends on the GraphType Parameter. If GraphType=VERTICAL the Data Legend will appear below the bottom axis. If GraphType=HORIZONTAL the Data Legend will appear to the left of the left axis.
Common CDL Attributes

**DataLegendGrid**

DataLegendGrid = (LineColor, bgColor, borderColor, bgImage, ImageFormat);

This parameter specifies the border and background colors for the Data Legend.

**Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LineColor</td>
<td>Specifies the color of the grid lines in the Data Legend.</td>
</tr>
<tr>
<td>BackgroundColor</td>
<td>Specifies the color for the Data Legend’s background.</td>
</tr>
<tr>
<td>BorderColor</td>
<td>Specifies an image file for the Data Legend’s background.</td>
</tr>
<tr>
<td>BackgroundImage</td>
<td></td>
</tr>
<tr>
<td>ImageFormat</td>
<td></td>
</tr>
</tbody>
</table>

**LineColor**

LineColor specifies the color of the grid lines in the Data Legend. See the common Color attribute in Chapter 4 for details.

**BackgroundColor**

BackgroundColor specifies the color for the Data Legend’s background. See the common Color attribute for details.

**BackgroundImage**

BackgroundImage specifies an image file for the Data Legend’s background. See the common Image attribute for details.

**DataLegendGridLine**

DataLegendGridLine = (LineType, LineStyle, LineWidth);

Defines the line properties for the Data Legend defined in the DataLegendGrid parameter, above.

**Used in These Charts**

Bar, Combo, Line
Example:

```
DataLegendGridLine = (HORIZONTAL, DOTTED, 2);
```

Attributes

<table>
<thead>
<tr>
<th>LineType</th>
<th>LineStyle</th>
<th>LineWidth</th>
</tr>
</thead>
</table>

**LineType**

Tells where to draw the lines specified with a `DataLegendGrid` parameter.

**Values**

- **BOTH**: draw both horizontal and vertical lines (default)
- **VERTICAL**: draw vertical lines only
- **HORIZONTAL**: draw horizontal lines only
- **NONE**: draw no grid lines

**Default**

BOTH

**LineStyle**

`LineStyle` tells how to draw the grid lines in a Data Legend as defined in a `DataLegendGrid` parameter.

**Values**

- **SOLID**: solid lines (default)
- **DOTTED**: dotted lines
- **DASHED**: dashed lines
- **DOTDASH**: dot-dash lines

**Default**

SOLID

**DataPointActiveLabels(n)**

```
DataPointActiveLabels(n) = ("Label1","URL1","Target1"),...
```

DataPointActiveLabels define the active labels associated with raw data points.

**Used in These Charts**

- Box Chart

**Example:**

```
DataPointActiveLabels = (lightgray, SHADOW, 3,,,gray);
```
Default
ON

Attributes
Color

DataPointColor

DataPointColor = Color;

DataPointColor allows users to specify the color to be used to display the raw data points. This value is used if no color is specified in the DataPointSymbol parameter.

Used in These Charts
Box Chart

Example:
DataPointColor = red;

Attributes
Color

DataPointJitter

DataPointJitter = ON | OFF;

DataPointJitter is used to increase the visibility of individual points when displaying raw data. The jitter option adds random horizontal jitter to the x values of each data point, allowing multiple points with the same Y value to be distinguishable.

Used in These Charts
Box Chart

Example:
DataPointJitter = ON;
DataPointJitter = OFF;
Default
ON

Attributes
Mode

DataPointSymbol

DataPointSymbol = (type1, size1, style1, bordercolor1, borderwidth1, image1, color1),...;

DataPointSymbol is used to define the style in which to draw data points when displaying raw data.

<table>
<thead>
<tr>
<th>typeN</th>
<th>the type of symbol to use for points in data series N. Legal values are NONE, CIRCLE, SQUARE, DIAMOND, CROSS, TARGET, TRIANGLEDOWN, TRIANGLEUP, IMAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>sizeN</td>
<td>the size in pixels of the symbols for points in data series N</td>
</tr>
<tr>
<td>styleN</td>
<td>the drawing style for points in data series N. Legal values are FILLED, OUTLINED or BOTH</td>
</tr>
<tr>
<td>borderColorN</td>
<td>the color of the border for points in data series N</td>
</tr>
<tr>
<td>borderWidthN</td>
<td>the width in pixels of the border for points in data series N</td>
</tr>
<tr>
<td>imageN</td>
<td>the image to use for displaying for points in data series N</td>
</tr>
<tr>
<td>colorN</td>
<td>the color for points in data series N</td>
</tr>
</tbody>
</table>

Used in These Charts
Box Chart

Example:

```
DataPointSymbol = (CIRCLE,3,FILLED,red,2,,green);
```

Attributes
BorderColor, BorderWidth, Color, Image, Type, Size, Style
**DataSet[n]**

DataSets[n] = a, b, c, ...;

DataSets[n] defines a list of numeric data values for each data set defined by the DataSets parameter in a charts having bars, such as bar, box, combo, pareto, strip, and time charts. Each data set may contain a different number of values. If the value, NULL, is substituted for a number, nothing will be drawn in that bar position.

**Used in These Charts**

Bar, Box, Combo, Pareto, Stock, Strip, Time

**Example:**

```plaintext
DataSets = ("Hamburgers", x33ffcc), ("Donuts", x9999ff), ("Cheez Doodles", xff6600), ("Exercise", xffcc33);
DataSets1 = 50, 12, 32, 52, 65, 40, 87;
DataSets2 = 40, 30, 77, 10, 25, 83, 9;
DataSets3 = 45, 50, 89, 33, 99, 44, 31;
DataSets4 = null, -4, -7, -12, -16, -23, -26;
```

**Attributes**

No attributes, as such, are used. Actual data appears in this parameter.

**DataSet[n]P[m]**

DataSets[n]P[m] = a, b, c, ...;

DataSets[n] defines a list of numeric values for each dataset in a grouped, stacked barchart (GraphType=GROUPSTACK). In a grouped stacked barchart each tic location can display multiple data sets, and each data set can contain multiple data values.

Consider this example:

```plaintext
DataSets = ("Set1"), ("Set2");
DataSets1P1 = 0.6, 0.7;
DataSets1P2 = 2.0, 1.1;
DataSets1P3 = 1.5, 2.0;
DataSets2P1 = 0.7, 0.9;
DataSets2P2 = 1.3, 2.1;
DataSets2P3 = 2.1, 1.4;
```

This chart contains two data sets; each set has 3 values to display at each tic on the chart.

DataSets1P1 defines two values - the first value in the first set at each tic mark. DataSets2P3 defines two values - the third value of the second set at each tic.
**DataSets**

\[ \text{DataSets}[N] = (\text{DataSetName}_1, \text{BorderType}_1, \text{BorderWidth}_1, \text{"ImageURL"}, \text{ImageFormat}_1, \text{BorderColor}_1), \ldots; \]

*DataSets* defines a list of data sets with the given name color and type for charts with bars in them, such as bar, box, combo, pareto, strip, and time charts. The Name attributes will be used as items in the legend; and the Color attribute will be used for each bar in the data set. The optional Type attribute indicates the shape of bar to use. *DataSets* must be paired with the corresponding *DataSet* parameter(s); you may specify from 1 up to 50 data sets.

**Used in These Charts**

Bar, Box, Combo, Pareto, Stock, Strip, Time

**Example:**

\[
\begin{align*}
\text{DataSets} & = ("Server \#1",",",","), ("Server \#2",",",","), ("Server \#3",",",","); \\
\text{DataSet1} & = 100, 125, 245.78, 147, 67; \\
\text{DataSet2} & = 85, 156, 179.5, 211, 123; \\
\text{DataSet3} & = 97, 87, 56, 267, 157;
\end{align*}
\]

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>Color</th>
<th>Type</th>
</tr>
</thead>
</table>

**Type**

Type indicates, in context of the *DataSets* parameter, the kind of bar that you see. Some Type values may produce unusual results if the GraphType is STACK or ROWS. If one of the pie types is selected, the alternate pie color will be extracted from the first element of a color table.

**Example:**

\[
\begin{align*}
\text{DataSets} & = ("Potatoes", \text{xC9933}, \text{DIAMONDBAR}), ("Green Beans", \text{darkgreen}, \text{BAR}), ("Tomatoes", \text{xC0033}, \text{CYLINDER}), ("Corn", \text{wheat}, \text{TRIANGLEBAR});
\end{align*}
\]

**Values**

- **BAR** A standard rectangular 3-D bar is shown
- **CYLINDER** The bars are 3-D, and the cross-section of the bars is circulars
- **DIAMONDBAR** The bars are 3-D, and the cross-section of the bars is a diamond shape
- **PIEHORIZONTAL** The bars are a horizontal pie shape
- **PIEVERTICAL** The bars are a vertical pie shape
- **TRIANGLEBAR** The bars are 3-D, and the cross-section of the bars is triangular.

**Default**

BAR

**DataType**

\[ \text{DataType}[N] = \text{Type}; \]

*DataSets* defines the type of data in the box chart’s data sets. This helps the chart to process the data.
Common CDL Attributes

Used in These Charts
Box

Example:

```
DataType = RAW;
DataType = SUMMARY;
```

Attributes
Type

Type indicates whether the data for the box chart has been statistically processed or not.

Example:

```
DataType = RAW;
DataType = SUMMARY;
DataSets = ("Sub-Compact", x00ab9c, BOX, 1),
          ("Compact", xf0887f),
          ("MiniVan", xf7bb83),
          ("Truck", x3fbae3),
          ("Luxury\nSedan", xf189af);
DataSet1 = 9.4, 10.2, 11.2, 7.5, 12.7, 22, 31;
DataSet2 = 10.4, 10.6, 10.8, 7.8, 13.5, 19;
DataSet3 = 17.6, 19.5, 24.0, 15.3, 30.5;
DataSet4 = 10.4, 11.0, 12.0, 7.8, 12.5;
DataSet5 = 25.6, 28.5, 33.0, 20.3, 35.5, 6;
```

Values

RAW
Data is raw statistical data. Percentiles and other ancillary information will be computed based on these values.

SUMMARY
Percentile information has already been computed. When a SUMMARY is used, the data is formatted as follows:
Value 1 25th percentile
Value 2 50th percentile
Value 3 75th percentile
Value 4 Smallest non-outlier value
Value 5 Largest non-outlier value
Value 6 + Outliers, if any

Default
None
Common CDL Attributes

DebugClear

DebugClear = debugFilter;

The **DebugClear** parameter clears the user selected debug information on the Java Console.

**Used in These Charts**

All

**Example:**

```plaintext
DebugClear = ALL;
```

**Attributes**

debugFilter

---

**DebugSet**

DebugSet = debugFilter;

The **DebugSet** parameter allows users to generate debug information on the Java Console for the following operations.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGENT</td>
<td>Agent status and errors.</td>
</tr>
<tr>
<td>ALL</td>
<td>No filtering, ALL debug information.</td>
</tr>
<tr>
<td>AXIS</td>
<td>Axis generation and rendering information.</td>
</tr>
<tr>
<td>BEANS</td>
<td>Loadable data object information.</td>
</tr>
<tr>
<td>CACHE</td>
<td>Internal memory cache status and error messages.</td>
</tr>
<tr>
<td>DWELL</td>
<td>Any issues relating to dwell (popup/active) labels.</td>
</tr>
<tr>
<td>FILE</td>
<td>File I/O and interpretation</td>
</tr>
<tr>
<td>GRAPH</td>
<td>Graph rendering status and errors</td>
</tr>
<tr>
<td>HTTP</td>
<td>HTTP Request/Response headers and status codes.</td>
</tr>
<tr>
<td>IMAGE</td>
<td>Image loading errors (e.g. attempting to load a background.png image for a chart.)</td>
</tr>
<tr>
<td>JDBC</td>
<td>Java DataBase Connectivity (JDBC) information including driver, connections, SQL, and result sets.</td>
</tr>
<tr>
<td>LEGEND</td>
<td>Legend generation and rendering.</td>
</tr>
<tr>
<td>LICENSE</td>
<td>License location and processing.</td>
</tr>
<tr>
<td>NOTES</td>
<td>Annotation generation and rendering information.</td>
</tr>
<tr>
<td>PARAM</td>
<td>Parameter parsing errors</td>
</tr>
<tr>
<td>REMOTE</td>
<td>Reports Named Data Set (NDS) processing information and errors.</td>
</tr>
<tr>
<td>SECURITY</td>
<td>User authentication information.</td>
</tr>
<tr>
<td>SERVER</td>
<td>Server specific issues, port requests, ACL, event scheduling, connection limits, and more.</td>
</tr>
</tbody>
</table>
The example below shows how to generate debug information on the Java Console for FILE operations, using the following syntax:

```
DebugSet = FILE;
DebugClear = FILE;
```

The DebugSet command enables the printing of debug messages for all subsequent FILE keywords, including the processing of all FileFormat Parameters. The debug messages will continue to be printed until the end of the Parameter script or the DebugClear directive is seen.

For example, the following script defines a simple piechart, enabling debug messages for all FILE statements:

```
DebugSet = FILE;
FileFormat = ("", ",", ",n", null, null, "/");
Slices = FILE "pietest.dat";
```

Resulting in the following debug output on the Java Console:

```
NFFile: startDelim = <>
NFFile: itemDelim = <|>
NFFile: endDelim = <\n>
NFFile: comments = <>
NFFile: Parameter = Slices
NFFile: Filename = <pietest.dat>
NFFile: Opening http://mycompany.com/reports/pietest.dat
NFFile: Item(1): 10|blue|Fred Smith NFFile: Item(2): 20|red|Sally Jane
NFFile: Item(3): 30|green|Oscar Jones
NFFile: Processed 3 Item(s)
NFFile: -----------------------
```

**Used in These Charts**

All

**Example:**

```
DebugSet = FILE;
```

**Attributes**

`debugFilter`

**DialActiveLabels**

```
DialActiveLabels[N] = (Name, Label, URL, Target), ...;
```

`DialActiveLabels` defines a list of active label destinations for a dial in a dial chart.
**Common CDL Attributes**

**Switch**

**Used in These Charts**

Dial

**Example:**

\[ \text{DialActiveLabels} = ("Destination", "demo.html", "frame1"); \]

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
</table>

**DialBorders**

**Example:**

\[ \text{DialBorders} = ("Hour Dial", \text{SOLID}, 2, \text{lightgray}, \text{CENTER}); \]

**Attributes**

<table>
<thead>
<tr>
<th>Name</th>
<th>LineType</th>
<th>LineWidth</th>
<th>Color</th>
<th>(Switch)</th>
</tr>
</thead>
</table>

**Switch**

This switch controls the dial border’s polygon behavior.

**Example:**

\[ \text{DialBorders} = ("Hour Dial", \text{SOLID}, 2, \text{lightgray}, \text{CENTER}); \]

**Values**

- **NONE**
  - The border doesn’t connect its ends

- **CENTER**
  - Includes the (circular) center of the dial in the borders

- **ENDTOEND**
  - Only includes the points in the border.

**Default**

No defaults

**DialClip**

\[ \text{DialClip} = \text{clipType}; \]
DialClip specifies quadrant(s) of the Dial Chart to be displayed using the entire graph space. For instance, specifying TOP will cause the dial to use the entire graph space to display the top-half of the dial. Specifying BOTTOMRIGHT will cause the dial to use the entire graph space to display only the bottom-right quadrant of the dial. The default is NONE which causes the entire dial to be displayed.

**Used in These Charts**
Dial

**Example:**
```plaintext
DialClip = TOP;
```

DialClip values can be:

<table>
<thead>
<tr>
<th>TOPLEFT</th>
<th>TOP</th>
<th>TOPRIGHT</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEFT</td>
<td>NONE</td>
<td>RIGHT</td>
</tr>
<tr>
<td>BOTTOMLEFT</td>
<td>BOTTOM</td>
<td>BOTTOMRIGHT</td>
</tr>
</tbody>
</table>

**DialClipPad**

DialClipPad = N;

DialClipPad specifies the margin to be used in conjunction with DialClip.

**Used in These Charts**
Dial

**Example:**
```plaintext
DialClipPad = 10;
```

**DialDelete**

DialDelete = (Name| ALL), ...;

DialDelete is used to delete a specific dial, or all dials, in a dial chart.

**Used in These Charts**
Dial

**Example:**
```plaintext
DialDelete = ("Hour Dial"), ("Minute Dial");
DialDelete = ALL;
```

**Attributes**
(Switch)
**Switch**

This switch allows you to either name the dials that are to be deleted, or to delete all at once.

**Example:**

```plaintext
DialDelete = ("Hour Dial"), ("Minute Dial");
DialDelete = ALL;
```

**Values**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A string that names a dial</td>
</tr>
<tr>
<td>ALL</td>
<td>All the dials</td>
</tr>
</tbody>
</table>

**Default**

No defaults

---

**DialFillPattern**

DialFillPattern = (name, type, color1, color2, imageURL), ...;

The DialFillPattern parameter provides a visual pattern fill inside the dial of a dial chart.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>no pattern, do default fill, if any</td>
</tr>
<tr>
<td>Built-In Patterns</td>
<td></td>
</tr>
<tr>
<td>FSLASH</td>
<td>front slash type</td>
</tr>
<tr>
<td>BSLASH</td>
<td>back slash type</td>
</tr>
<tr>
<td>DGRID</td>
<td>diagonal grid lines, (front and back slash lines)</td>
</tr>
<tr>
<td>HORIZONTAL</td>
<td>horizontal lines</td>
</tr>
<tr>
<td>VERTICAL</td>
<td>vertical lines</td>
</tr>
<tr>
<td>GRID</td>
<td>grid lines, (horizontal and vertical lines)</td>
</tr>
<tr>
<td>Gradients</td>
<td></td>
</tr>
<tr>
<td>GRADIENTVERTICAL</td>
<td>bottom to top style gradient</td>
</tr>
<tr>
<td>GRADIENTHORIZONTAL</td>
<td>left to right style gradient</td>
</tr>
<tr>
<td>GRADIENTFDIAG</td>
<td>top right to bottom left style gradient</td>
</tr>
<tr>
<td>GRADIENTBDIAG</td>
<td>top left to bottom right style gradient</td>
</tr>
<tr>
<td>GRADIENTRADIAL</td>
<td>radial style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERHORIZONTAL</td>
<td>center out horizontal style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERVERTICAL</td>
<td>center out vertical style gradient</td>
</tr>
<tr>
<td>Images</td>
<td></td>
</tr>
<tr>
<td>IMAGE</td>
<td>use an image specified in the optional imageURL element</td>
</tr>
</tbody>
</table>

**color1**

This color is used in the following ways:
- Foreground color for patterns
- Starting color for gradients
- Ignored in images
**Used in These Charts**

Dial

**Example:**

```
DialFillPattern = ("Name", GRADIENTHORIZONTAL, blue, white);
```

**Attributes**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Color1</th>
<th>Color2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageFormat</td>
<td>ImageURL</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**DialFills**

```
DialFills = (Name, Color, NONE\|CENTER\|ENDTOEND), ...
```

**DialFills** controls the decorative fill inside a dial in a dial chart. Its attributes are standard area attributes, except for the switch.

**Used in These Charts**

Dial

**Example:**

```
DialFills = ("Hour Dial", xebf0f3, CENTER);
```

**Attributes**

<table>
<thead>
<tr>
<th>Name</th>
<th>Color</th>
<th>(Switch)</th>
</tr>
</thead>
</table>

**Switch**

This switch controls the dial border's polygon behavior.

**Example:**

```
DialFills = ("Hour Dial", xebf0f3, CENTER);
```
Common CDL Attributes

Values

NONE The border doesn’t connect its ends
CENTER Includes the (circular) center of the dial in the borders
ENDTOEND Only includes the points in the border.

Default
No defaults

DialFormats

DialFormats = (Name, FLOAT|INTEGER|DECIMAL, formatExpression), ...;

DialFormats allow for the formatting of dial labels.

Used in These Charts
Dial

Example:

DialFormats = ("Hour Dial", FLOAT, %.2f);

Attributes
Name (Switch) formatExpression

Switch

This switch controls the dial format type. If the format type is INTEGER or FLOAT, the input data value is expected to be of type integer or float and will be parsed as such (if string conversion is necessary). The format itself is a C-language style sprintf format. Some examples:

<table>
<thead>
<tr>
<th>Data</th>
<th>Type</th>
<th>Format</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>INTEGER</td>
<td>%d</td>
<td>1000</td>
</tr>
<tr>
<td>1000</td>
<td>INTEGER</td>
<td>$,%d</td>
<td>$1,000</td>
</tr>
<tr>
<td>1000</td>
<td>INTEGER</td>
<td>%d%</td>
<td>1000%</td>
</tr>
<tr>
<td>1000</td>
<td>FLOAT</td>
<td>%f</td>
<td>1000.0</td>
</tr>
<tr>
<td>1000</td>
<td>FLOAT</td>
<td>%.2f</td>
<td>1000.00</td>
</tr>
<tr>
<td>1000</td>
<td>FLOAT</td>
<td>$,.2f</td>
<td>$1,000.00</td>
</tr>
</tbody>
</table>

If the format type is DECIMAL, the format syntax is consistent with those defined in the Java DecimalFormat spec.

Default
FLOAT
Common CDL Attributes

DialHandAnimationStyle

DialHandAnimationStyle = GROW | FADE | NONE

Specifies how the dial hands initially appear in a dial chart. This parameter is only valid in SVG or SVGWeb output formats.

Attributes

Style

Style refers to the manner in which dial hands are first rendered in a dial chart.

Example:

DialHandAnimationStyle = GROW;

Values

GROW The dial hands rise from a diameter of zero to their actual values.
FADE The dial hands fade in.
NONE The dial hands are immediately visible.

Default

NONE

Dials

Dials = (Name, StartAngle, StopAngle, RadiusPercentage, NONE|INSIDE|OUTSIDE), . . . ;

The Dials parameter, essential to dial charts, arranges the appearance for a set of uniquely named dials within a dial chart. There may be more than one dial in a dial chart, and they may overlap. For example, an analog clock can be considered to have three overlapping dials: hour, minute, and second. Dials may also be arranged next to each other in concentric circles. Dials are layers one atop the other, the first one defined being on the bottom and subsequent dials above it, overlapping.

Used in These Charts

Dial

Example:

Dials = ("Hour Dial", 0, 360, 100, INSIDE),
("Minute Dial", 0, 360, 100, INSIDE),
("Second Dial", 0, 360, 100, NONE);
<table>
<thead>
<tr>
<th>Attributes</th>
<th>Name</th>
<th>StartAngle</th>
<th>StopAngle</th>
<th>RadiusPercentage</th>
<th>(Switch)</th>
</tr>
</thead>
</table>

**StartAngle** indicates the angle from which the dial starts, going counter-clockwise from the vertical. This parameter, along with **StopAngle**, allows you to create dials that occupy less than a full circle.

**Example:**

```
Dials = ("Internal Pressure", -135, 135, 100, INSIDE),
       ("Atmospheric Pressure", -135, 135, 60, INSIDE),
       ("Pressure Change", -135, 135, 30, INSIDE);
```

**Values**

Numerical degrees from 0 to + or -360

**Default**

No defaults

**StopAngle** indicates the angle at which the dial stops, going counter-clockwise from the vertical. This parameter, along with **StartAngle**, allows you to create dials that occupy less than a full circle.

**Example:**

```
Dials = ("Hour Dial", 0, 360, 100, INSIDE),
       ("Minute Dial", 0, 360, 100, INSIDE),
       ("Second Dial", 0, 360, 100, NONE);
```

**Values**

Numerical degrees from 0 to + or -360

**Default**

No defaults
RadiusPercentage

RadiusPercentage controls the width of the dial, in the radial direction, that the dial occupies in the chart. This parameter allows you to create concentric dials.

Example:

Dials = ("Hour Dial",0,360,100,INSIDE),
       ("Minute Dial",0,360,100,INSIDE),
       ("Second Dial",0,360,100,NONE);

Dials = ("Internal Pressure",-135,135,100,INSIDE),
       ("Atmospheric Pressure",-135,135,60,INSIDE),
       ("Pressure Change",-135,135,30,INSIDE);

Dials = ("Dial",-180,180,100,INSIDE);

Values
Numerical percentage from 0 to 100

Default
No defaults

Switch

This switch controls where the tics fall on the dial’s face.

Example:

Dials = ("Hour Dial",0,360,100,INSIDE),
       ("Minute Dial",0,360,100,INSIDE),
       ("Second Dial",0,360,100,NONE);

Dials = ("Internal Pressure",-135,135,100,INSIDE),
       ("Atmospheric Pressure",-135,135,60,INSIDE),
       ("Pressure Change",-135,135,30,INSIDE);

Dials = ("Dial",-180,180,100,INSIDE);

Values
NONE No tics are shown
INSIDE Tics appear within the dial’s face
OUTSIDE Tics appear outside (but adjacent to) the dial’s face

Default
No defaults

DialScale

DialScale = (Name, MinValue, MaxValue, StepValue);
The DialScale parameter specifies the minimum and maximum data values that will be displayed for the named dials in a dial chart. If the DialScale parameter is not defined, or the MinValue and MaxValue parameters are the same or one of them is not defined, then the tic mark locations will be automatically determined based on the actual data values being displayed. That is, the dial will be "autoscaled" using the current data values to determine "reasonable" values for MinValue, MaxValue and StepValue. If values are supplied for any of MinValue, MaxValue, or StepSize, those values will be used as part of the autoscaling.

**Used in These Charts**

Dial

**Example:**

DialScale = ("Hour Dial", 0, 12, 1), ("Minute Dial", 0, 60, 1), ("Second Dial", 0, 60, 1);

**Attributes**

<table>
<thead>
<tr>
<th>Name</th>
<th>MinValue</th>
<th>MaxValue</th>
<th>StepValue</th>
</tr>
</thead>
</table>

**MinValue**

MinValue sets the absolute lower visible limit for the dial scale.

**Example:**

DialScale = ("Hour Dial", 0, 12, 1), ("Minute Dial", 0, 60, 1), ("Second Dial", 0, 60, 1);

DialScale = ("Internal Pressure", 0, 10000, 1000), ("Atmospheric Pressure", 10, 100, 1), ("Pressure Change", 0, 1000, 100);

**Values**

Any real number less than MaxValue

**Default**

None

**MaxValue**

MaxValue sets the absolute upper visible limit for the dial scale.

**Example:**

DialScale = ("Hour Dial", 0, 12, 1), ("Minute Dial", 0, 60, 1), ("Second Dial", 0, 60, 1);

DialScale = ("Internal Pressure", 0, 10000, 1000), ("Atmospheric Pressure", 10, 100, 1), ("Pressure Change", 0, 1000, 100);

**Values**

Any real number greater than MinValue

**Default**

None
**StepValue**

`StepValue` is optional, and may be used to specify a given step between tic marks along the bottom axis, starting with the `MinValue`. If `StepValue` is not an even multiple of the difference between the `MinValue` and `MaxValue`, then no tic mark will be displayed at the `MaxValue`. If `StepValue` is not defined, tic marks will be placed at "reasonable" locations along the axis, depending on the range of values being displayed.

**Example:**

```plaintext
DialScale = ("Internal Pressure",0,10000,1000),("Atmospheric Pressure",10,100,1),("Pressure Change",0,1000,100);
```

**Values**

Any real number between `MinValue` and `MaxValue`

**Default**

1

**DialSectorAnimationStyle**

`DialSectorAnimationStyle = GROW | FADE | NONE`

Specifies how the dial sectors initially appear in a dial chart. This parameter is only valid in SVG or SVGWeb output formats.

**Attributes**

`Style`

**Style**

`Style` refers to the manner in which dial sectors are first rendered in a dial chart.

**Example:**

```plaintext
DialSectorAnimationStyle = GROW;
```

**Values**

- **GROW**: The dial sectors rise from a diameter of zero to their actual values.
- **FADE**: The dial sectors fade in.
- **NONE**: The dial sectors are immediately visible.

**Default**

`NONE`
DialSize

DialSize = (minWidth, minHeight, maxWidth, maxHeight);

The DialSize parameter can be used to set minimum and maximum sizes for the actual dial in a dial chart. This allows programmers to guarantee that the dial will always be the same size regardless of the length of the strings in the legend or tic labels. DialSize has the following interaction with DialSquare; if the minimum or maximum dimensions specified are not square, and DialSquare is ON, then the dial will be inscribed in a square with a dimension that ranges from the smallest of the minimum values to the smallest of the maximum values.

_used in These Charts_

Dial

_example_

DialSize = (100,100,200,200);

Attributes

minWidth, minHeight, maxWidth, maxHeight

DialSquare

DialSquare = mode;

The DialSquare parameter tells the dial chart that the appearance of the dial should be kept as high as it is wide.

_used in These Charts_

Dial

_example_

DialSquare = ON|OFF;

Attributes

Mode

DialTicLabels

DialTicLabels = (Name, Label1, Label2, ..., LabelN), . . .;

DialTicLabels allows you to label the tic marks for a named dial in a dial chart.
**Common CDL Attributes**

<table>
<thead>
<tr>
<th>Name</th>
<th>Label</th>
</tr>
</thead>
</table>

**Label**

Label can be one of several strings that identify the tic marks around a dial in a dial chart. This is equivalent to putting numbers around the face of a clock.

**Example:**

```
DialTicLabels = ("Hour Dial", "12", "1", "2", "3", "4", "5", "6", "7", "8",
                 "9", "10", "11");
DialTicLabels = ("Internal Pressure", "0", "1000", "2000", "3000", "4000",
                 "5000", "6000", "7000", "8000", "9000", "10,000");
```

**Values**

Use any string. If there are more labels than tics, only the labels that correspond to tic marks will be shown. If there are fewer labels than tic marks, all labels will be shown.

**Default**

None

**DialTicLabelStyles**

```
DialTicLabelStyles = (Name, ON|OFF, LabelPos, Color, FontName, FontSize, Angle, interiorAlignment), ...
```

DialTicLabelStyles controls the appearance of the tic mark labels defined with the DialTicLabels parameter.

<table>
<thead>
<tr>
<th>interiorAlignment</th>
<th>Specifies the alignment to use in text strings that contain multiple lines.</th>
</tr>
</thead>
</table>

The legal values for interiorAlignment are LEFT, RIGHT, or CENTER.

**Used in These Charts**

Dial
Common CDL Attributes

Switch

This switch turns the tic labels’ visibility on and off.

Example:

DialTicLabelStyles = ("Hour Dial","ON",1.1,black,"Helvetica",14,0),
"Minute Dial","ON",1.1,black,"Helvetica",14,0);
**Common CDL Attributes**

<table>
<thead>
<tr>
<th>Attributes</th>
<th>PercentofRadius</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td><strong>Color</strong></td>
</tr>
<tr>
<td>Dial</td>
<td></td>
</tr>
</tbody>
</table>

**Example:**

DialTics = ("Hour Dial",gray,1,10),("Minute Dial",gray,1,5);

DialTics = ("Internal Pressure",black,2,5),
           ("Internal Pressure",green,2,5),
           ("Pressure Change",red,1,4);

**Attributes**

- **PercentofRadius** controls the length of the tics as a percentage of the dial's radius.

**Example:**

DialTics = ("Hour Dial",gray,1,10),("Minute Dial",gray,1,5);

DialTics = ("Internal Pressure",black,2,5),
           ("Internal Pressure",green,2,5),
           ("Pressure Change",red,1,4);

**Values**

Percentages from 0 to 100

**Default**

No defaults

**DrawFences**

DrawFences = ON | OFF;

The **DrawFences** parameter specifies whether or not to draw fences on the chart. The default is ON.

**Used in These Charts**

Box

**Example:**

DrawFences = OFF;

**Attributes**

- **Mode**
**DrawOrder**

\[
\text{DrawOrder}[N] = (\text{SYMBOL}, \ldots);
\]

The `DrawOrder` parameter defines the order in which Bar, Line, LineFill and Stock options will be drawn.

- **BAR** refers to ALL bar sets
- **LINE** refers to line sets which do NOT have a fill
- **LINEFILL** refers to line sets which DO have a fill
- **STOCK** refers to ALL stock sets

**Used in These Charts**

Combo, Pareto, Stock

**Example:**

\[
\text{DrawOrder} = (\text{BAR});
\]

**Attributes**

**Symbol**

**DwellAnimationHighlightBorderStyle**

\[
\text{DwellAnimationHighlightBorderStyle} = (\text{lineType}, \text{LineWidth}, \text{lineColor});
\]

Defines the border style to be applied to a datapoint when `DwellAnimationStyle` = HIGHLIGHT.

**Used in These Charts**

All

**Example:**

\[
\text{DwellAnimationHighlightBorderStyle} = (\text{DASHED}, 1, \text{BLACK});
\]

**Attributes**

**LineType**

**LineWidth**

**LineColor**
### Common CDL Attributes

#### LineType

<table>
<thead>
<tr>
<th>Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOLID</td>
<td>A solid line is displayed.</td>
</tr>
<tr>
<td>DOTTED</td>
<td>A dotted line is displayed.</td>
</tr>
<tr>
<td>DASHED</td>
<td>A dashed line is displayed.</td>
</tr>
<tr>
<td>DOTDASH</td>
<td>Alternating dots and dashes are displayed.</td>
</tr>
</tbody>
</table>

#### Default

SOLID

---

### DwellAnimationHighlightFill

DwellAnimationHighlightFill = \texttt{Color};

Defines the color used to fill a datapoint when \texttt{DwellAnimationStyle = HIGHLIGHT}.

---

### Used in These Charts

All

---

### Example:

DwellAnimationHighlightFill = \texttt{BLUE});

---

### Attributes

Color

---

### Default

NONE

---

### DwellAnimationStyle

DwellAnimationStyle = \texttt{HIGHLIGHT | NONE}

Defines how the chart behaves when the mouse dwells over a data point. DwellAnimation parameters are only valid for image output types (i.e. DwellAnimation does not use the applet supported SVG and SVGweb output formats).

---

### Example:

DwellAnimationStyle = \texttt{HIGHLIGHT};
Values

HIGHLIGHT The data point is highlighted using the values specified in DwellAnimationHighlightFill and DwellAnimationHighlightBorderStyle.

NONE No highlight is applied to the data point.

Default NONE

DwellBox

\[
DwellBox[N] = (\text{Color}, \text{BorderType}, \text{BorderWidth}, "\text{ImageURL}", \text{ImageFormat}, \text{BorderColor}, \text{TRCornerStyle}, \text{BRCornerStyle}, \text{BLCornerStyle}, \text{CornerColor});
\]

The DwellBox parameter is optional. If specified, it defines a box to be displayed as a background for each dwell label specified by a DwellLabel parameter. The box will be automatically scaled to fit each dwell label.

Used in These Charts

All

Example:

\[
DwellBox = (\text{yellow}, \text{RAISED}, 3);
\]

Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color</td>
<td>The color of the box.</td>
</tr>
<tr>
<td>BorderType</td>
<td>The border style of the box.</td>
</tr>
<tr>
<td>BorderWidth</td>
<td>The width of the box border.</td>
</tr>
<tr>
<td>ImageURL</td>
<td>The URL of the image to be displayed.</td>
</tr>
<tr>
<td>ImageFormat</td>
<td>The format of the image.</td>
</tr>
<tr>
<td>BorderColor</td>
<td>The color of the border.</td>
</tr>
<tr>
<td>TRCornerStyle</td>
<td>The drawing style for the top right corner.</td>
</tr>
<tr>
<td>BRCornerStyle</td>
<td>The drawing style for the bottom right corner.</td>
</tr>
<tr>
<td>BLCornerStyle</td>
<td>The drawing style for the bottom left corner.</td>
</tr>
<tr>
<td>CornerColor</td>
<td>The color of the corners.</td>
</tr>
</tbody>
</table>

XXCornerStyle

The drawing style for each of the four corners of the region. Styles are specified in a clockwise fashion starting in the upper left - Top Left, Top Right, Bottom Right, and Bottom Left. Legal values are SQUARE, SNIP and ROUND. The default is SQUARE.

DwellLabel

\[
DwellLabel[N] = ("\text{Label}", \text{Color}, "\text{FontName}", \text{FontSize}, \text{Angle}, \text{interiorAlignment});
\]
If the DwellLabel parameter is defined, then a label will automatically be displayed whenever the mouse cursor dwells over a given data value. The attributes defined for the DwellLabel parameter specify the format of each label, not its text value.

| interiorAlignment | Specifies the alignment to use in text strings that contain multiple lines. |

The legal values for interiorAlignment are LEFT, RIGHT, or CENTER.

**Used in These Charts**

All

**Example:**

```
DwellLabel = ("", black, "Courier", 12, LEFT);
```

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>Color</th>
<th>FontName</th>
<th>FontSize</th>
</tr>
</thead>
</table>

**DwellOffset**

```
DwellOffset = size;
```

The DwellOffset specifies the length of the sides the define the square area around a "hotspot" associated with an active label. When the mouse moves into this hotspot, an active label, if defined, will be displayed. When the mouse is clicked within this hotspot, the target URL, if defined, will be "drilled" to. By default, $\text{DwellOffset} = 20$, which means the hotspot for a datapoint will be a 20 by 20 pixel square centered over the data point. Set DwellOffset to something smaller to prevent overlap between the hotspots of tightly packed datapoints.

If you are using a symbol then the symbol size is added on to the dwell hotspot size to compute the dwell offset area.

**Used in These Charts**

Line, XY

**Example:**

```
DwellOffset = 5;
```

**Attributes**

<table>
<thead>
<tr>
<th>size</th>
</tr>
</thead>
</table>

© Visual Mining, Inc 2002-2011
www.visualmining.com
Edges

Edges[N] = (NodeStart, NodeEnd, Color, Direction, LineStyle, LineWidth, ArrowStyle, ArrowLength, ArrowWidth), ...;

Within a diagram chart, the Edges parameter lists and defines the appearance of the lines to be drawn between named nodes.

**Used in These Charts**

Diagram

**Example:**

Edges = ("node1","node2",black,NONE,SOLID,3,BLOCK,15,8);

Edges = ("server","client",black,FROMTO,SOLID,1,BLOCK,9,4);

Edges = ("CEO", "coo-node", dimgray, NONE, SOLID, 1),
("coo-node", "sales-node", dimgray, NONE, SOLID, 1),
("coo-node", "marketing-node", dimgray, NONE, SOLID, 1),
("sales-node", "VP Sales", dimgray, NONE, SOLID, 1),
("coo-node", "COO", dimgray, NONE, SOLID, 1),
("marketing-node", "VP Marketing", dimgray, NONE, SOLID, 1),
("VP Marketing", "Webmaster", dimgray, NONE, SOLID, 1);

**Attributes**

<table>
<thead>
<tr>
<th>NodeStart</th>
<th>NodeEnd</th>
<th>Color</th>
<th>Direction</th>
<th>LineStyle</th>
</tr>
</thead>
<tbody>
<tr>
<td>LineWidth</td>
<td>ArrowStyle</td>
<td>ArrowLength</td>
<td>ArrowWidth</td>
<td></td>
</tr>
</tbody>
</table>

**NodeStart**

NodeStart names the node where the edge begins.

**Example:**

Edges = ("node1","node2",black,NONE,SOLID,3,BLOCK,15,8);

Edges = ("server","client",black,FROMTO,SOLID,1,BLOCK,9,4);

Edges = ("CEO", "coo-node", dimgray, NONE, SOLID, 1),
("coo-node", "sales-node", dimgray, NONE, SOLID, 1),
("coo-node", "marketing-node", dimgray, NONE, SOLID, 1),
("sales-node", "VP Sales", dimgray, NONE, SOLID, 1),
("coo-node", "COO", dimgray, NONE, SOLID, 1),
("marketing-node", "VP Marketing", dimgray, NONE, SOLID, 1),
("VP Marketing", "Webmaster", dimgray, NONE, SOLID, 1);

**Values**

The identifying string, in double-quotes, of a node named with the Nodes parameter.

**Default**

No defaults
**NodeEnd**

NodeEnd names the node where the edge ends.

**Example:**

```
Edges = ("node1","node2",black,NONE,SOLID,3,BLOCK,15,8);
Edges = ("server","client",black,FROMTO,SOLID,1,BLOCK,9,4);
Edges = ("CEO", "coo-node", dimgray, NONE, SOLID, 1),
       ("coo-node", "sales-node", dimgray, NONE, SOLID, 1),
       ("coo-node", "marketing-node", dimgray, NONE, SOLID, 1),
       ("sales-node", "VP Sales", dimgray, NONE, SOLID, 1),
       ("coo-node", "COO", dimgray, NONE, SOLID, 1),
       ("marketing-node", "VP Marketing", dimgray, NONE, SOLID, 1),
       ("VP Marketing", "Webmaster", dimgray, NONE, SOLID, 1);
```

**Values**

The identifying string, in double-quotes, of a node named with the Nodes parameter.

**Default**

No defaults

**Direction**

Direction the direction of the arrow head(s), if any, for the edge.

**Example:**

```
Edges = ("server","client",black,FROMTO,SOLID,1,BLOCK,9,4);
Edges = ("CEO", "coo-node", dimgray, NONE, SOLID, 1),
       ("coo-node", "sales-node", dimgray, NONE, SOLID, 1),
       ("coo-node", "marketing-node", dimgray, NONE, SOLID, 1),
       ("sales-node", "VP Sales", dimgray, NONE, SOLID, 1),
       ("coo-node", "COO", dimgray, NONE, SOLID, 1),
       ("marketing-node", "VP Marketing", dimgray, NONE, SOLID, 1),
       ("VP Marketing", "Webmaster", dimgray, NONE, SOLID, 1);
```

**Values**

- **NONE**  - No Arrows are shown
- **FROMTO**  - Arrow from NodeStart node to NodeEnd node
- **TOFROM**  - Arrow from NodeEnd to NodeStart node
- **BOTH**  - Arrow in both directions

**Default**

FROMTO
**ArrowStyle**

ArrowStyle, if used, determines the style of the arrowhead(s) on the edge. This attribute is optional.

**Example:**

```
Edges = ("node1","node2",black,NONE,SOLID,3,BLOCK,15,8);
Edges = ("server","client",black,FROMTO,SOLID,1,BLOCK,9,4);
```

**Values**

- **NONE**: No Arrows are shown
- **SHARP**: Triangular tip with sharper corners than BLOCK
- **ROUND**: Circular tip
- **BLOCK**: Triangular tip
- **LINE**: Plain line tip

**Default**

BLOCK

**ArrowLength**

ArrowLength determines the length, in pixels, from the start to the tip of the arrow(s) on an edge. This attribute is optional.

**Example:**

```
Edges = ("node1","node2",black,NONE,SOLID,3,BLOCK,15,8);
Edges = ("server","client",black,FROMTO,SOLID,1,BLOCK,9,4);
```

**Values**

Whole pixel numbers

**Default**

No defaults

**ArrowWidth**

ArrowWidth determines the width, in pixels, of the arrow(s) on an edge. This attribute is optional.

**Example:**

```
Edges = ("node1","node2",black,NONE,SOLID,3,BLOCK,15,8);
Edges = ("server","client",black,FROMTO,SOLID,1,BLOCK,9,4);
```

**Values**

Whole pixel numbers
**Default**
No defaults

**EightyLineSetName**

EightyLineSetName = *name*;

Name assigned to the 80% line. Used in the legend if the Legend CDL parameter does not define a label for the 80% line.

*Used in These Charts*
Pareto

*Example:*

EightyLineSetName = “80% Line”

**EightyTwentyLineStyle**

EightyTwentyLineStyle = *(Type, LineWidth, Color, FillColor, LineType)*;

This parameter specifies the line style to be displayed for the 80/20 lines.

*Used in These Charts*
Pareto

*Example:*

EightyTwentyLineStyle = *(SOLID, 3, blue, blue, NORMAL)*;

**Attributes**

<table>
<thead>
<tr>
<th>Color</th>
<th>FillColor</th>
<th>LineType</th>
<th>LineWidth</th>
<th>Type</th>
</tr>
</thead>
</table>

**FillColor**

If this attribute is not NULL, then the area under the 80/20 lines be filled with the given color.

*Example:*

EightyTwentyLineStyle = *(SOLID, 3, red, pink, NORMAL)*;

<!-- red line with pink fill -->

*Values*

<table>
<thead>
<tr>
<th>NULL</th>
<th>Also, value left unspecified: No color fills the area under the line.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any legal color</td>
<td>Area under the line is filled. See Chapter 4 for the Color attribute.</td>
</tr>
</tbody>
</table>

**Default**
None
| **Type** |  
| --- | --- |
| The style of line to draw |  
| **Values** |  
| NONE |  
| SOLID |  
| DOTTED |  
| DASHED |  
| DOTDASH |  
| **Default** |  
| SOLID |  
| **LineType** |  
| The type of line to use to connect the points in the line set |  
| **Values** |  
| NORMAL |  
| FIT |  
| BOTH |  
| **Default** |  
| NORMAL |  
| **EightyTwentyLineSymbol** |  
| EightyTwentyLineSymbol = (Type, Size, Style, BorderColor, BorderWidth, ImageURL, SymbolColor, ShadowWidth); |  
| EightyTwentyLineSymbol specifies the symbols to be displayed for the 80/20 lines. |  
| **Used in These Charts** |  
| Pareto |  
| **Example:** |  
| EightyTwentyLineSymbol = (CIRCLE, 6, BOTH, white, l, grey, 0); |  
| **Attributes** |  
| BorderColor | BorderWidth | ImageURL | ShadowWidth |  
| Size | Style | SymbolColor | Type |
**Size**

Size specifies the size of the symbol in pixels. This attribute is ignored for IMAGE symbols.

*Example:*

EightyTwentyLineSymbol = (CIRCLE, 6, BOTH, white, 1);

*Values*

Any integer value in pixels

*Default*

None

**SymbolColor**

SymbolColor specifies the foreground color of the EightyTwentyLineSymbol.

**ShadowThickness**

ShadowThickness specifies size of the shadow behind a EightyTwentyLineSymbol. Any number other than 0 causes NetCharts to choose a shadow size based on the size of the symbol. Specifying a value of 0 suppresses the shadow.

**Style**

Style specifies how the EightyTwentyLineSymbol should be drawn, including FILLED, OUTLINED, or BOTH. If FILLED is specified, the symbol is filled with the line set color. If OUTLINED is specified, only the outline is drawn, using the line set color. If BOTH is specified, then the symbol is filled with the line set color and the outline is drawn using the borderColor.

*Example:*

EightyTwentyLineSymbol = (CIRCLE, 6, BOTH, white, red);

*Values*

- FILLED: Symbol is filled with the cumulative line color.
- OUTLINED: Only the outline is drawn, using the cumulative line color.
- BOTH: Symbol is filled with the 80/20 line color and the outline is drawn using the borderColor

*Default*

None

**Type**

Type specifies the type of symbol to be displayed on the line set.
Example:

\[
\text{EightyTwentyLineSymbol} = (\text{SQUARE}, 6, \text{BOTH}, \text{cyan}, 1);
\]

Values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>No symbol is displayed.</td>
</tr>
<tr>
<td>CIRCLE</td>
<td>Displays circles</td>
</tr>
<tr>
<td>SQUARE</td>
<td>Displays squares</td>
</tr>
<tr>
<td>DIAMOND</td>
<td>Displays diamonds</td>
</tr>
<tr>
<td>CROSS</td>
<td>Displays crosses</td>
</tr>
<tr>
<td>TARGET</td>
<td>Displays targets (bulls-eye)</td>
</tr>
<tr>
<td>TRIANGLEDOWN</td>
<td>Displays downward pointing triangles</td>
</tr>
<tr>
<td>TRIANGLEUP</td>
<td>Displays upward pointing triangles</td>
</tr>
<tr>
<td>IMAGE</td>
<td>If specified, the ImageURL attribute is required and will be used to load a GIF image for the symbol.</td>
</tr>
</tbody>
</table>

Default

None

FenceActiveLabels

\[
\text{FenceActiveLabels}_N = (\text{"Label1"}, \text{"URL1"}, \text{"Target1"}),...;
\]

FenceActiveLabels\_N define the active labels associated with fences in data set \( N \).

Used in These Charts

- Box Chart

Example:

\[
\text{FenceActiveLabels} = (\text{"LabelText"},,);
\]

Attributes

Label, Target, URL

FencePosition

\[
\text{FencePosition} = \text{Type};
\]

FencePosition specifies whether to draw the fences that are within the Inter Quartile Range, (the box), over or under the box.

Used in These Charts

- Box Chart
Example:
FencePosition = UNDER;
FencePosition = OVER;

Attributes
Type

Type refers to the fences that are within the Inter Quartile Range, (the box).

Values

<table>
<thead>
<tr>
<th></th>
<th>Draw fences under the box</th>
<th>Draw fences over the box</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNDER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OVER</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Default
OVER

FontEncoding

The FontEncoding parameter refers to the "codepage" that should be used when mapping fonts. Some examples of FontEncodings are UTF-8, cp1252, cp850, iso 8859. This rarely needs to be changed, even when non-ASCII fonts are being used.

Footer

Footer = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment, exteriorAlignment);

Footer, which is universal to NetCharts applets, describes an optional title, or label, that sits at the bottom of a chart, or at its “foot,” and uses standard attributes for string text, text color, font, font size, and label rotation.

Used in These Charts
All

Example:
Footer = ("This Is A\nMulti-Line\nFooter", darkred);

interiorAlignment | Specifies the alignment to use in text strings that contain multiple lines.
Common CDL Attributes

<table>
<thead>
<tr>
<th>attributeName</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>exteriorAlignment</td>
<td>Specifies the alignment for the entire Title object.</td>
</tr>
</tbody>
</table>

The legal values for `interiorAlignment` and `exteriorAlignment` are LEFT, RIGHT, or CENTER.

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>Color</th>
<th>FontName</th>
<th>FontSize</th>
<th>Angle</th>
<th>interiorAlignment</th>
<th>exteriorAlignment</th>
</tr>
</thead>
</table>

**FooterActiveLabel**

`FooterActiveLabel = ("Label", "URL", "Target");`

`FooterActiveLabel` defines a single active label destination for the footer title.

**Used in These Charts**

All

**Example:**

`FooterActiveLabel = ("Where To Go", "demo.html", "frame1");`

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
</table>

**FooterBox**

`FooterBox = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);`

The `FooterBox` specifies a background region just for the chart footer title.

**Used in These Charts**

All

**Attributes**

<table>
<thead>
<tr>
<th>BorderColor</th>
<th>BorderType</th>
<th>BorderWidth</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageFormat</td>
<td>ImageURL</td>
<td>TRCornerStyle</td>
<td>BRCornerStyle</td>
</tr>
<tr>
<td>BLCornerStyle</td>
<td>CornerColor</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**XXCornerStyle**

The drawing style for each of the four corners of the region. Styles are specified in a clockwise fashion starting in the upper left - Top Left, Top Right, Bottom Right, and Bottom Left. Legal values are SQUARE, SNIP and ROUND. The default is SQUARE.
**Format**

TopFormat[N] = (FormatType, “FormatExpr”, “TimeBase”, “TimeUnit”);
BottomFormat[N] = (FormatType, “FormatExpr”, “TimeBase”, “TimeUnit”);
LeftFormat[N] = (FormatType, “FormatExpr”, “TimeBase”, “TimeUnit”);
RightFormat[N] = (FormatType, “FormatExpr”, “TimeBase”, “TimeUnit”);

Format adjusts the numeric labels that are automatically generated for the given axis, should one be defined.

**Used in These Charts**
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**
TopFormat = (DATE, "%w\n%M/%D", "1 Jan 2000", "1h");
TopFormat = (DATE, "%M/%D", "1 Apr 96", "1d");
TopFormat = (INTEGER);
TopFormat = (FLOAT, "$%,9.2f",,);

**Attributes**

<table>
<thead>
<tr>
<th>FormatType</th>
<th>FormatExpr</th>
<th>TimeBase</th>
<th>TimeUnit</th>
</tr>
</thead>
</table>

**FormatType** specifies the type of number being displayed on the given axis.

**Example:**
LeftFormat = (INTEGER, "$%f",,);

**Values**

DATE Axis value are shown as date and/or time values. See Appendix A: Date and Time Values for further detail.
FLOAT Axis values are shown with decimal parts.
INTEGER Axis values are shown as integers, and are rounded if necessary.
DECIMAL Axis values are shown as decimals, see [http://java.sun.com/j2se/1.3/docs/api/java/text/DecimalFormat.html](http://java.sun.com/j2se/1.3/docs/api/java/text/DecimalFormat.html) for more information.

The characters listed here are used in non-localized patterns. Localized patterns use the corresponding characters taken from this formatter's DecimalFormatSymbols object instead, and these characters lose their special status. Two exceptions are the currency sign and quote, which are not localized.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Location</th>
<th>Localized?</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Number</td>
<td>Y</td>
<td>Digit</td>
</tr>
<tr>
<td>#</td>
<td>Number</td>
<td>Y</td>
<td>Digit, zero shows as absent</td>
</tr>
<tr>
<td>.</td>
<td>Number</td>
<td>Y</td>
<td>Decimal separator or monetary decimal separator</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Y</td>
<td><strong>Common CDL Attributes</strong></td>
</tr>
<tr>
<td>---</td>
<td>--------</td>
<td>----</td>
<td>---------------------------</td>
</tr>
<tr>
<td><code>-</code></td>
<td>Number</td>
<td>Y</td>
<td>Minus sign</td>
</tr>
<tr>
<td><code>,</code></td>
<td>Number</td>
<td>Y</td>
<td>Grouping separator</td>
</tr>
<tr>
<td><code>E</code></td>
<td>Number</td>
<td>Y</td>
<td>Separates mantissa and exponent in scientific notation. <em>Need not be quoted in prefix or suffix.</em></td>
</tr>
<tr>
<td><code>;</code></td>
<td>Subpattern boundary</td>
<td>Y</td>
<td>Separates positive and negative subpatterns</td>
</tr>
<tr>
<td><code>%</code></td>
<td>Prefix or suffix</td>
<td>Y</td>
<td>Multiply by 100 and show as percentage</td>
</tr>
<tr>
<td><code>\u2030</code></td>
<td>Prefix or suffix</td>
<td>Y</td>
<td>Multiply by 1000 and show as per mille</td>
</tr>
<tr>
<td><code>\x (\u00A4)</code></td>
<td>Prefix or suffix</td>
<td>N</td>
<td>Currency sign, replaced by currency symbol. If doubled, replaced by international currency symbol. If present in a pattern, the monetary decimal separator is used instead of the decimal separator.</td>
</tr>
<tr>
<td><code>,'</code></td>
<td>Prefix or suffix</td>
<td>N</td>
<td>Used to quote special characters in a prefix or suffix, for example, &quot;##&quot; formats 123 to &quot;#123&quot;. To create a single quote itself, use two in a row: &quot;# o'clock&quot;.</td>
</tr>
</tbody>
</table>

**Default**

**INTEGER**

**TimeBase**

The **TimeBase** attribute specifies the base date to be used when determining the actual date or time value when using a time unit or numeric value. See **Appendix A: Date and Time** Values for further detail.

**Example:**

```
LeftFormat = (INTEGER, "%f", "10");
```

**Values**

String values representing dates or times

**Default**

None

**TimeUnit**

The **TimeUnit** attribute controls the time multiplier to be used when determining the actual data/time value when using a numeric value. See **Appendix A: Date and Time** Values for further detail.
**Example:**

```
TopFormat = (DATE, "%w\n%M/%D", "1 Jan 2000", "1h");
TopFormat = (DATE, "%M/%D", "1 Apr 96", "1d");
```

**Values**

String values representing dates or times

**Default**

None

---

**GraphLayout**

```
GraphLayout = Type;
```

*GraphLayout* defines the bar orientation in a chart.

**Used in These Charts**

Bar, Box, Combo, Line, Stock,

**Example:**

```
GraphLayout = HORIZONTAL;
GraphLayout = VERTICAL;
```

**Attributes**

*Type*

**Type**

```
Type
```

*Type* refers to the direction the bars lie in the graph.

**Example:**

```
GraphLayout = HORIZONTAL;
GraphLayout = VERTICAL;
```

**Values**

- **HORIZONTAL** The bar sets are oriented running from left to right.
- **VERTICAL** The bar sets are oriented rising from bottom to top.

**Default**

**VERTICAL**
**GraphType**

GraphType = Type;

GraphType defines the type of multiple-line graph to be displayed, and mostly affects how the stacking is achieved.

**Used in These Charts**
Bar, Combo, Line, Stock,

**Type**

Type refers to the manner in which lines or bars are stacked in a chart.

**Example:**
GraphType = STACK;

**Values**

**GROUPSTACK** For bar series only, sets are collected together in groups at each tic, and at each tic a group member has a stack of values.

**ROWS** The bar or line sets are displayed separately. If 3DDepth is nonzero, then they will be displayed in separate rows, from front to back.

**STACK** The bar or line sets are stacked on top of each other. That is, as each data set is drawn, its values are added to previous values displayed. Negative and NULL values are treated as zero.

**PERCENT** The bar or line sets are stacked on top of each other, normalized to 100 percent. That is, as each data set is drawn, its values are added to previous values displayed and displayed as a percentage of the total of all values. Negative and NULL values are treated as zero.

**Default**

STACK

**Grid**

Grid = (LineColor1, bgColor1, borderColor1, bgImage1, ImageFormat1),...;

All Visual Mining charts, except for the Dial chart, Diagram chart, and Pie chart support the display of one or more grids behind the data. The grid layouts and styles can be independently specified, and can be associated with any of the axes being displayed. The Grid parameter allows you to specify up to three grid sets.

Since all of the Grid parameters are defined as vectors, you can specify more than one grid for the same chart. This allows arbitrary combinations of styles and spacing to achieve a wide array of chart grids.

A NULL background color can be assigned to second and subsequent grids so that it will not overwrite the first grid. However, looking at the second example below, if one didn’t specify a NULL background
color, then white would have been used as the default and the second grid would completely overwrite the first.

**Used in These Charts**
Bar, Box, Bubble, Combo, Line, Pareto, Radar, Stock, Strip, Time, X-Y

**Example:**
```
Grid = (green, white, black, "../images/mychartbg.gif", CENTER);
<br>   <!-- The following produces a striped grid -->
Grid = (green, white), (black, null);
GridLine = (VERTICAL, BAR), (HORIZONTAL, DOTTED, 2);
```

**Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LineColor</td>
<td>Specifies the color of the grid lines. See the common Color attribute in Chapter 4 for details.</td>
</tr>
<tr>
<td>BackgroundColor</td>
<td>Specifies the color for the grid’s background. See the common Color attribute for details.</td>
</tr>
<tr>
<td>BackgroundImage</td>
<td>Specifies an image file for the grid’s background. See the common Image attribute for details.</td>
</tr>
</tbody>
</table>

**LineColor**

LineColor specifies the color of the grid lines. See the common Color attribute in Chapter 4 for details.

**BackgroundColor**

BackgroundColor specifies the color for the grid’s background. See the common Color attribute for details.

**BackgroundImage**

BackgroundImage specifies an image file for the grid’s background. See the common Image attribute for details.

**Grid3DDepth**

```
Grid3DDepth = depth;
```

In any chart that can display bar data sets, the Bar3DDepth parameter defines the depth of each bar in pixels, as well as the depth of the underlying grids.

In all cases, the grids defined in the Grid parameters automatically adjust to the current depth setting.

**Used in These Charts**
Bar, Box, Bubble, Combo, Line, Pareto, Stock, Strip, Time, X-Y
**Example:**

```
Grid3DDepth = 13;  <!-- displays a 3-D grid 13 pixels deep -->
```

**Attributes**

**Depth**

If the `Depth` parameter is set to 0, then 2D bars and grids are displayed. If that parameter is not defined, then a default depth will be chosen.

**Example:**

```
Grid3DDepth = 10;  <!-- displays a 3-D grid 10 pixels deep -->
Grid3DDepth = 0;   <!-- displays a 2-D grid -->
```

**Values**

Depth in pixels

**Default**

For charts with 3-D bars, a depth commensurate with the bar depth is chosen automatically.

---

**GridAnimationStyle**

```
GridAnimationStyle = FADE | NONE
```

Defines how grid backgrounds initially appear in a chart. This parameter is only valid in SVG or SVGweb output formats.

**Example:**

```
GridAnimationStyle = FADE;
```

**Values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FADE</td>
<td>The grids fade in.</td>
</tr>
<tr>
<td>NONE</td>
<td>The grids are immediately visible.</td>
</tr>
</tbody>
</table>

**Default**

NONE

---

**GridAxis**

```
GridAxis = (XAxis1, YAxis1),...;
```

`GridAxis` specifies pairs of X and Y axes for each grid set in the chart, matching the sets in the `Grid` parameter. By default, grids use the bottom and left axes to determine the spacing of the grid lines.
GridAxis parameter allows you to specify arbitrary combinations of axes for each grid being displayed. This allows you to easily display grids with different types of lines spaced at various intervals. GridAxis is commonly used in conjunction with the Grid and GridLine parameters.

**Used in These Charts**
Bar, Box, Bubble, Combo, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**

```
Grid = (green, white), (black, null);
GridLine = (HORIZONTAL, SOLID, 2), (HORIZONTAL, DOTTED, 1);
GridAxis = (BOTTOM, LEFT), (BOTTOM, RIGHT);
```

**Attributes**

<table>
<thead>
<tr>
<th>XAxis</th>
<th>YAxis</th>
</tr>
</thead>
</table>

---

**GridBlockActiveLabels**

```
GridBlockActiveLabels = ("Label1", "URL1", "Target1"),...;
```

GridBlockActiveLabels The GridBlockActiveLabels parameter specifies a list of custom active labels to be associated with each grid block. The labels will be displayed whenever the mouse "dwell" over a given grid block being displayed.

**Used in These Charts**
Heat Map

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
</table>

---

**GridBlockBackgroundColor**

```
GridBlockBackgroundColor = color...;
```

GridBlockBackgroundColor specifies the default grid block background color.

**Used in These Charts**
Heat Map

**Attributes**

<table>
<thead>
<tr>
<th>Color</th>
</tr>
</thead>
</table>

**Default**
White
GridBlockCellColorType

GridBlockCellColorType = type;

   GridBlockCellColorType determines the cell color type.

Used in These Charts
Heat Map

Attributes
Type

Example:

GridBlockCellColorType=COLORTABLE;

Values
COLORTABLE
EXPRESSIONS
SPECTRUM

GridBlockColors

GridBlockColors = (color1, color2,...,colorN);

   If too few colors are specified for the grid blocks the color pattern repeats.

Used in These Charts
Heat Map

Attributes
Color

GridBlockColorSpectrum

GridBlockColorSpectrum = (color1,color2,min,max,gradientstep);

   GridBlockColorSpectrum defines attributes for a color spectrum. A color spectrum is generated from two colors which are the starting and ending colors and from the number of steps which is the number of color buckets. The minimum and maximum values represent the data range for the entire spectrum. Each color bucket in the spectrum represents a certain data range. If a data value of a grid block falls within that range, it will be shown with its associated color. The spectrum colors start at color1, go to white or
almost white, depending on the number of steps, and then end at color2. The number of steps determines the amount each color bucket is changed by and the data values each color represents.

*Used in These Charts*

- Heat Map

**Attributes**

<table>
<thead>
<tr>
<th>Color1</th>
<th>Color2</th>
<th>Min</th>
<th>Max</th>
<th>gradientstep</th>
</tr>
</thead>
</table>

**Example**

```
GridBlockColorSpectrum = (black,grey,2,45,20);
```

**GridBlockExpressions**

```
GridBlockExpressions = ("operator",value1,value2,color),...;
```

Each color expression has an operator, value(s) to compare against, and a color to use if the expression is true. The BETWEEN operator is the only operator that is used to compare 2 values.

*Used in These Charts*

- Heat Map

**Attributes**

<table>
<thead>
<tr>
<th>Operator</th>
<th>value1</th>
<th>value2</th>
<th>color</th>
</tr>
</thead>
</table>

**Operator**

Specifies the operator used

**Values**

">", "<", ">=", "<="", "==", "!="", BETWEEN

**Example**

```
GridBlockExpressions = ("BETWEEN",2,9,yellow),("===",1,aqua)...;
```

**GridBlockLabel**

```
GridBlockLabel = ("mode", color, "font name", font size);
GridBlockTopLabel = ("mode", color, "font name", font size, angle, interiorAlignment);
GridBlockLeftLabel = ("mode", color, "font name", font size, angle, interiorAlignment);
GridBlockRightLabel = ("mode", color, "font name", font size, angle, interiorAlignment);
GridBlockBottomLabel = ("mode", color, "font name", font size, angle, interiorAlignment);
```
Defines the text style for the grid block labels

**Used in These Charts**
Heat Map

**Attributes**

<table>
<thead>
<tr>
<th>mode</th>
<th>font name</th>
<th>font size</th>
<th>angle</th>
<th>interiorAlignment</th>
</tr>
</thead>
</table>

**Example:**

```
GridBlockLeftLabel = ("ON",purple,"ARIAL",12,0,CENTER);
```

**Default**

*ON*

**GridBlockLabels**

```
GridBlockLabels = label1,label2,...,labelN;
```

```
GridBlockTopLabels = label1,label2,...,labelN;
GridBlockBottomLabels = label1,label2,...,labelN;
GridBlockLeftLabels = label1,label2,...,labelN;
GridBlockRightLabels = label1,label2,...,labelN;
```

Defines the list of labels to center over the grid blocks

**Used in These Charts**
Heat Map

**Attributes**

*Label*

**Example**

```
GridBlockLabels = “Derek”,“Joe”,“Reggie”;```

GridBlockLayout

GridBlockLayout = (Height, Width);
Defines the size of grid. If not defined the grid attempts to layout in a square. The maximum grid size is 50 x 50

Used in These Charts
Heat Map

Attributes
Height Width

Example

GridBlockLayout=(2,7);

GridBlockLine

GridBlockLine = ("LineStyle", width, color);
Defines the line style to be displayed in the grid.

Used in These Charts
Heat Map

Attributes
Style Width Color

LineStyle

LineStyle tells how to draw the grid line.

Example:
GridBlockLine="Solid", 1, pink;

Values
SOLID solid lines
DOTTED dotted lines
DASHED dashed lines
DOTDASH dot-dash lines
**GridBlockSort**

GridBlockSort = (SortType, SortOrder);

Determines the sort order for the grid values.

*Used in These Charts*

Heat Map

*Attributes*

<table>
<thead>
<tr>
<th>SortType</th>
<th>SortOrder</th>
</tr>
</thead>
</table>

*SortType*

The item that the grid block will be sorted on.

*Example:*

GridBlockSort = (Label, ASCENDING);

*Values*

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
</tr>
<tr>
<td>LABEL</td>
</tr>
<tr>
<td>VALUE</td>
</tr>
</tbody>
</table>

*SortOrder*

*Values*

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCENDING</td>
</tr>
<tr>
<td>DESCENDING</td>
</tr>
</tbody>
</table>

**GridBlockValues**

GridBlockValues = Value1, Value2, Value3 ...

Defines a list values for the heat map dataset

*Used in These Charts*

Heat Map
**Attributes**

Value

---

**GridBlockValueFormat**

GridBlockValueFormat = (FormatType, "FormatExpr")

The GridBlockValueFormat parameter defines the format for the grid block values in the heat map.

**Used in These Charts**

Heat Map

**Example:**

GridBlockValueFormat = (DECIMAL, "%2");

**Attributes**

FormatType FormatString

**FormatType**

The type of number to use when formatting the values.

**Values**

FLOAT
INTEGER
DECIMAL

---

**GridBlockValueStyle**

GridBlockValueStyle = ("mode", color, "font name", font size);

Defines styles for grid values.

**Used in These Charts**

Heat Map

**Attributes**

mode font name font size
**Common CDL Attributes**

**LineType**

*Example*

```plaintext
GridBlockValueStyle = ("ON",purple,"ARIAL",12);
```

**Default**

ON

**GridLine**

```plaintext
GridLine = (LineType,LineStyle,LineWidth),... ;
```

One or more grid line styles can be specified using the `GridLine` parameter. Each set of parameters defines the line properties for the corresponding grid defined in the `Grid` parameter, above.

**Used in These Charts**

Bar, Box, Bubble, Combo, Line, Pareto, Stock, Strip, Time, X-Y

*Example:*

```plaintext
GridLine = (VERTICAL, BAR), (HORIZONTAL, DOTTED, 2);
```

**Attributes**

<table>
<thead>
<tr>
<th>LineType</th>
<th>LineStyle</th>
<th>LineWidth</th>
</tr>
</thead>
</table>

**LineType**

Tells where to draw the lines, is specified with a `GridLine` parameter, and refers to a `Grid` parameter.

*Example:*

```plaintext
GridLine = (VERTICAL, BAR), (HORIZONTAL, DOTTED, 2);
```

**Values**

<table>
<thead>
<tr>
<th>BOTH</th>
<th>draw both horizontal and vertical lines (default)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VERTICAL</td>
<td>draw vertical lines only</td>
</tr>
<tr>
<td>HORIZONTAL</td>
<td>draw horizontal lines only</td>
</tr>
<tr>
<td>NONE</td>
<td>draw no grid lines</td>
</tr>
</tbody>
</table>

**Default**

BOTH

**LineStyle**

LineStyle tells how to draw the grid lines, is specified with a `GridLine` parameter, and refers to a `Grid` parameter.
### Example:

```
GridLine = (VERTICAL, BAR), (HORIZONTAL, DOTTED, 2);
```

### Values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOLID</td>
<td>solid lines (default)</td>
</tr>
<tr>
<td>DOTTED</td>
<td>dotted lines</td>
</tr>
<tr>
<td>DASHED</td>
<td>dashed lines</td>
</tr>
<tr>
<td>DOTDASH</td>
<td>dot-dash lines</td>
</tr>
<tr>
<td>BAR</td>
<td>alternating bars instead of lines</td>
</tr>
</tbody>
</table>

**Default**: SOLID

### GroupStackLabels

```
GroupStackLabels = Label, Label, ...;
```

This parameter is used only for grouped stacked barcharts (GraphType=GROUPSTACK). It defines labels for each stack of data points in a group. If this value is unspecified and ShowGroupStackLabels=ON; the set names defined in the DataSets parameter will be used as the stack labels.

### GroupStackSegmentLabels

```
GroupStackSegmentLabels = Label, Label, ...;
```

This parameter is used only for grouped stacked barcharts (GraphType=GROUPSTACK). It defines labels for each value in a stack of data points at a single tic location. These labels will be displayed in the legend of the chart. GroupStackSegmentLabels may be overridden by the LegendItems parameter.

### HandActiveLabels

```
HandActiveLabels[N] = (Name, Label, URL, Target), ...;
```

Specifies sets of active labels attached to the hands in a dial chart. Each grouped set in parenthesis, or “tuple,” has a corresponding set within a DataSet parameter.

**Used in These Charts**

Dial

**Example:**

```
HandActiveLabels = ("Minute", "Minute", "DialChartMin.html", "InfoFrame"),
                 ("Hour", "Hour", "DialChartHr.html", InfoFrame),
                 ("Second", "Second", "DialChartSec.html", InfoFrame);
```

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
</table>
HandBorders

HandBorders = (Name, LineType, LineWidth, Color);

HandBorders specifies the line style to apply to the borders of dial hands. The default line color is black.

Used in These Charts

Dial

Example:

HandBorders = ("Hand1", DASHED, 2, DarkGray), ("Hand2", DASHED, 2, DarkGray);

Attributes

<table>
<thead>
<tr>
<th>HandName</th>
<th>LineType</th>
<th>LineWidth</th>
<th>Color</th>
</tr>
</thead>
</table>

LineType

LineType specifies the style of the border to be drawn on a chart’s hands.

Values

<table>
<thead>
<tr>
<th>LineType</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOLID</td>
<td>Draws a solid line of LineWidth thickness.</td>
</tr>
<tr>
<td>DOTTED</td>
<td>Draws a dotted line of LineWidth thickness.</td>
</tr>
<tr>
<td>DASHED</td>
<td>Draws a dashed line of LineWidth thickness.</td>
</tr>
<tr>
<td>DOTDASH</td>
<td>Draws a dot-dashed line of LineWidth thickness.</td>
</tr>
</tbody>
</table>

Default

SOLID

LineWidth

LineWidth specifies the width in pixels of the border to be drawn on a chart’s hands.

Values

<table>
<thead>
<tr>
<th>LineWidth</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 or greater</td>
<td>Whole number width in pixels</td>
</tr>
</tbody>
</table>

Default

1

Color

Color specifies the color of the hand border.
HandButtonBorder

HandButtonBorder = (LineType, LineWidth, Color);

HandButtonBorder specifies the line style to apply to the center button of a dial. The default line color is black.

Used in These Charts

Dial

Example:

HandButtonBorder = (SOLID, 2, DarkGray);

Attributes

<table>
<thead>
<tr>
<th>LineType</th>
<th>LineWidth</th>
<th>Color</th>
</tr>
</thead>
</table>

LineType

LineType specifies the style of the border to be drawn on a chart’s hands.

Values

<table>
<thead>
<tr>
<th>LineType</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOLID</td>
<td>Draws a solid line of LineWidth thickness.</td>
</tr>
<tr>
<td>DOTTED</td>
<td>Draws a dotted line of LineWidth thickness.</td>
</tr>
<tr>
<td>DASHED</td>
<td>Draws a dashed line of LineWidth thickness.</td>
</tr>
<tr>
<td>DOTDASH</td>
<td>Draws a dot-dashed line of LineWidth thickness.</td>
</tr>
</tbody>
</table>

Default

SOLID

LineWidth

LineWidth specifies the width in pixels of the border to be drawn on a chart’s hands.

Values

<table>
<thead>
<tr>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 or greater</td>
<td>Whole number width in pixels</td>
</tr>
</tbody>
</table>

Default

1

Color

Color specifies the color of the hand border.
**HandData**

\[
\text{HandData}[N] = (\text{Name}, \text{Value}, \text{Length}), \ldots;
\]

- **HandData** identifies and describes hands in dial charts.

**Used in These Charts**

Dial

**Example:**

\[
\text{HandData} = (\text{"Hour Hand"}, 3.5, 68), (\text{"Minute Hand"}, 30, 85), (\text{"Second Hand"}, 53, 95);
\]

**Attributes**

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Length</th>
</tr>
</thead>
</table>

- **Value** gives the value of the hand relative to the dial's range.

**Example:**

\[
\text{HandData} = (\text{"Hour Hand"}, 3.5, 68), (\text{"Minute Hand"}, 30, 85), (\text{"Second Hand"}, 53, 95);
\]

**Values**

Real numbers

**Default**

None

**Length**

- **Length** gives the length of the hand as a percentage of the dial's radius.

**Example:**

\[
\text{HandData} = (\text{"Hour Hand"}, 3.5, 68), (\text{"Minute Hand"}, 30, 85), (\text{"Second Hand"}, 53, 95);
\]

**Values**

Real numbers

**Default**

None

**HandDelete**

\[
\text{HandDelete}[N] = (\text{Name}\{|\text{ALL}\}, \ldots);
\]

- **HandDelete** is used to delete a specific hand, or all hands, in a dial chart.
Common CDL Attributes

Switch

Used in These Charts
Dial

Example:

```
HandDelete = ("Hour"), ("Minute");
HandDelete = ALL;
```

Attributes
(Switch)

Switch

This switch allows you to either name the hands that are to be deleted, or to delete all at once.

Example:

```
HandDelete = ("Pressure"), ("PressureChange");
HandDelete = ALL;
```

Values

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A string that names a hand</td>
</tr>
<tr>
<td>ALL</td>
<td>All the hands</td>
</tr>
</tbody>
</table>

Default

No defaults

HandDrag

```
HandDrag[N] = "ON" | "OFF";
```

The HandDrag switch is used to allow or stop the user from dragging the hands of the dial chart with the mouse.

Used in These Charts
Dial

Example:

```
HandDrag = "ON";
HandDrag = "OFF";
```

Attributes
(Switch)

Switch

This switch sets the on/off state.
Example:

```java
HandDrag = "ON";
HandDrag = "OFF";
```

### Values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON</td>
<td>Allows the user to drag the hands on the applet dial</td>
</tr>
<tr>
<td>OFF</td>
<td>Stops the user from dragging the hands on the applet dial</td>
</tr>
</tbody>
</table>

### Default

```
OFF
```

---

**HandDropShadow**

```
HandDropShadow = (color, offsetx, offsety, size);
```

`HandDropShadow` places a shadow on the background of a hand in a dial chart. The color, orientation, and size of the shadow can be defined. The tuple element `color` sets the color of the shadow. That color value is interpolated to complete transparency as it reaches the end of the shadow’s blur area. `Offsetx` and `offsety` define the center point of the shadow; `offsetx` sets the x-axis offset from the chart’s center-point; `offsety` sets the y-axis offset. When an offset attribute is set to a whole number value, the position of the drop shadow is offset from the chart’s center point by the number of pixels set by that whole number. When an offset is set to a fractional value (between 0 and 1), the value is interpreted as a percentage of the width of the object casting the shadow (the set of bars). The center of the drop shadow is repositioned based upon the values or percentages set for `offsetx` and `offsety`. Offset attribute values may be positive or negative. `Size` sets the size of the blur area, the region beyond the actual drop shadow shape where the shadow is extended and blurred into transparency. The size of this blurred region is controlled by the `size` attribute. The blurred region becomes larger and more diffuse as the value of `size` is increased. When `size` is set to a whole number value, the size of the blurred area is defined in pixels. When `size` is set to a fractional value (between 0 and 1), the value is interpreted as a percentage of the width of the object casting the shadow. When using a fractional value, enclose the value in double-quotes to “escape” the decimal point.

### Used in These Charts

Dial

### Example:

```java
HandDropShadow = (color, offsetx, offsety, size);
```

### Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color</td>
<td></td>
</tr>
<tr>
<td>Offsetx</td>
<td></td>
</tr>
<tr>
<td>offsety</td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td></td>
</tr>
</tbody>
</table>

### Color

```
Color specifies the base color of the shadow to be drawn behind a chart’s bars.
```
**Offsetx**

Offsetx specifies the x-coordinate offset from center.

**Offsety**

Offsety specifies the y-coordinate offset from center.

**Size**

Size specifies the width of the blurred area.

**HandButtonEdgeHighlights**

HandButtonEdgeHighlights = (start,stop,gap,size), ...;

The HandButtonEdgeHighlights parameter provides a visual pattern fill in a Dial chart which accents the center button. It overlays a ring (annulus) fill pattern over the existing fill patterns in a specified zone along the interior edge of the center button. The gap between the sides of the center button and the fill pattern being applied can be modified. The element start sets the beginning color of the gradient, associated with the outside edge; the element stop sets the end color of the gradient, associated with the interior of the center button where the color blends to transparency. Color values are interpolated between the two. The element size specifies the width of the highlight. The element gap specifies the size of the gap between the edge of the highlight and the edge of the center button. When the value for gap is specified as a whole number, it sets the distance between the edge of the highlight and the edge of the center button in pixels. When set to a fractional number between 0 and 1, it sets the gap to a percentage of the radius of the pie. Percentage values are written using a decimal point (e.g. “0.05” and “0.50” represent 5% and 50%, respectively). When using a fractional value, enclose the tuple element in double-quotes to “escape” the decimal point.

**Used in These Charts**

Dial

**Example:**

HandButtonEdgeHighlights = (blue_25,white_75,1,25), ...;

**Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td></td>
</tr>
<tr>
<td>stop</td>
<td></td>
</tr>
<tr>
<td>gap</td>
<td></td>
</tr>
<tr>
<td>size</td>
<td></td>
</tr>
</tbody>
</table>
Hands

Hands[N] = (Name, TipColor, ShaftColor, DialName, HandLabel), ...;

The Hands parameter names the hands of a dial chart, colors them, and links them to a dial.

Used in These Charts
Dial

Example:

   Hands = ("Hour Hand", black, black, "Hour Dial","Hours"),
           ("Minute Hand", black, black, "Minute Dial"),
           ("Second Hand", xAC0000, xAC0000, "Second Dial");

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>TipColor</th>
<th>ShaftColor</th>
<th>DialName</th>
</tr>
</thead>
</table>

TipColor

TipColor controls the color of the hand’s tip.

Example:

   Hands = ("Hour Hand", black, black, "Hour Dial"),
           ("Minute Hand", black, black, "Minute Dial"),
           ("Second Hand", xAC0000, xAC0000, "Second Dial");

Values

Use values for the regular Color attribute as described in Chapter 8.

Default
black

ShaftColor

ShaftColor controls the color of the hand’s shaft.

Example:

   Hands = ("Hour Hand", black, black, "Hour Dial"),
           ("Minute Hand", black, black, "Minute Dial"),
           ("Second Hand", xAC0000, xAC0000, "Second Dial");

Values

Use values for the regular Color attribute as described in Chapter 8.

Default
black

DialName

DialName specifies which dial within the dialchart to which the hand is associated. (There can be more than one dial, and they can overlap physically.)
Example:

\[
\text{Hands} = (\text{"Hour Hand"}, \text{black}, \text{black}, \text{"Hour Dial"}), \\
(\text{"Minute Hand"}, \text{black}, \text{black}, \text{"Minute Dial"}), \\
(\text{"Second Hand"}, \text{xAC0000}, \text{xAC0000}, \text{"Second Dial"});
\]

Values

Use values for the regular Name attribute as described in Chapter 8.

Default

None given

HandLabel

HandLabel specifies an optional text label to display along with the hand. The label’s appearance is controlled by the HandLabels parameter.

HandLabels

\[
\text{HandLabels} = (\text{"Name"}, \text{ON|OFF}, \text{labelpos}, \text{Color}, \text{"FontName"}, \text{FontSize}, \text{Angle}, \text{interiorAlignment});
\]

HandLabels controls the appearance of text labels on the hands of a DialChart.

Used in These Charts

Dial

Example:

\[
\text{HandLabels} = (\text{"Goal"}, \text{ON\text{.1}}, \text{red}, \text{"Helvetica"}, \text{12}, \text{0}), \\
(\text{"Actual"}, \text{ON\text{.1}}, \text{black}, \text{"Helvetica"}, \text{12}, \text{0});
\]

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>(Switch)</th>
<th>LabelPos</th>
<th>Color</th>
<th>FontName</th>
</tr>
</thead>
<tbody>
<tr>
<td>FontSize</td>
<td>Angle</td>
<td>interiorAlignment</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Switch

This switch turns the hand labels’ visibility on and off.

Values

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ON</td>
<td>Show the hand label</td>
</tr>
<tr>
<td>OFF</td>
<td>Hide the hand label</td>
</tr>
</tbody>
</table>

Default

None
**InteriorAlignment**

The `LabelPos` attribute specifies the alignment to use in hand labels that contain multiple lines. The legal values for `interiorAlignment` are `LEFT`, `RIGHT`, or `CENTER`.

**LabelPos**

The `LabelPos` attribute in the dial chart works the same way the `LabelPos` parameter does for the pie chart: it defines the position of the tic mark labels relative to the width of the dial. A value of 1.1 will place each label just outside the dial face, while a value of 0.6 will place each label inside of the dial face. For instance, clocks are often made with their tic mark labels on the outside of the dial face, but gauges tend to have them just on the inside of the dial face along with the tic marks.

**Values**

Positive real numbers, generally between 0 and 2.

**Default**

None

**HandStyles**

HandStyles[N] = (Name, NEEDLELINE|NEEDLEFILL|NEEDLEBUTTON|SHARP|ROUND|BLOCK|LINE|NONE, TipWidth, ShaftWidth), ...;

The `HandStyles` parameter describes the appearance of the hands in a dial chart.

**Used in These Charts**

Dial

**Example:**

HandStyles = ("Hour Hand", BLOCK, 8, 4), ("Minute Hand", BLOCK, 6, 3),
            ("Second Hand", NONE, 4, 2);

**Attributes**

Name            (Switch)    TipWidth    ShaftWidth

**Switch**

This switch identifies the style of the tip of a hand in a dial chart.

**Example:**

HandStyles = ("Hour Hand", BLOCK, 8, 4), ("Minute Hand", BLOCK, 6, 3),
            ("Second Hand", NONE, 4, 2);
Common CDL Attributes

**Values**

<table>
<thead>
<tr>
<th>Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHARP</td>
<td>Triangular tip with sharper corners than BLOCK</td>
</tr>
<tr>
<td>ROUND</td>
<td>Circular tip</td>
</tr>
<tr>
<td>BLOCK</td>
<td>Triangular tip</td>
</tr>
<tr>
<td>LINE</td>
<td>Plain line tip</td>
</tr>
<tr>
<td>NONE</td>
<td>Hand will not be drawn</td>
</tr>
</tbody>
</table>

**Default**

SHARP

**Header**

```
Header = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment, exteriorAlignment, extend);
```

Header, which is universally available in Visual Mining chart applications, describes an optional title, or label, that sits at the visual top of a chart, or its “head,” and uses standard attributes for string text, text color, font, font size, and label rotation. Note that the Header is displayed as centered across the entire applet or graphic space, not across the grid of the chart. As this can sometimes look awkward, we recommend using the TopAxisTitle parameter instead if you want a header centered across the chart instead.

Header uses a standard CDL label grouping, or “tuple,” to describe the typography of this label.

**Used in These Charts**

All

**Example:**

```
Header = ("Jets Per Minute", crimson, Helvetica, 12, 0, LEFT, CENTER, OFF);
```

<table>
<thead>
<tr>
<th>interiorAlignment</th>
<th>Specifies the alignment to use in text strings that contain multiple lines.</th>
</tr>
</thead>
<tbody>
<tr>
<td>exteriorAlignment</td>
<td>Specifies the alignment for the entire Title object.</td>
</tr>
<tr>
<td>extend</td>
<td>Specified background should extend entire length of chart.</td>
</tr>
</tbody>
</table>

The legal values for interiorAlignment and exteriorAlignment are LEFT, RIGHT, or CENTER. The legal values for extend are ON and OFF.

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>Color</th>
<th>FontName</th>
<th>FontSize</th>
<th>Angle</th>
<th>interiorAlignment</th>
<th>exteriorAlignment</th>
</tr>
</thead>
</table>

**HeaderActiveLabel**

```
HeaderActiveLabel = ("Label", "URL", "Target");
```

HeaderActiveLabel defines a single active label destination for the header title.
**Used in These Charts**
All

**Example:**
```
HeaderActiveLabels = ("Destination", "demo.html", "frame1");
```

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
</table>

**HeaderBox**

```
HeaderBox = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor);
```

The **HeaderBox** specifies a background region just for the **Header** title.

**Used in These Charts**
All

**Attributes**

```
BorderColor  BorderType  BorderWidth  Color
ImageFormat   ImageURL   
```

**HistogramBin**

```
HistogramBin = (HistogramBinType, HistogramBinSize);
```

**HistogramBin** describes the type and number of bins in a histogram.

**Used in These Charts**
Histogram

**Example:**
```
HistogramBin = (BYNUMBER,3);
```

**Attributes**

```
HistogramBinType  HistogramBinSize
```

**HistogramBinType**

String constant describing the distribution type. Legal values are AUTO, BYNUMBER, BYWIDTH. The default value is AUTO.

AUTO allows the histogram to automatically define the bins.
BYNUMBER allows users to specify the number of bins.

BYWIDTH allows the user to specify the width of a bin.

**HistogramBinSize**

Integer specifying the histogram bin size. If HistogramBinType is AUTO, this field is ignored. If HistogramBinType is BYNUMBER, this field specifies the number of bins. If HistogramBinType is BYWIDTH this field specifies the width of each bin.

**HistogramScale**

HistogramScale = (HistogramMinValue, HistogramMaxValue);

HistogramScale defines the upper and lower limits of the data range of a histogram.

**Used in These Charts**

Histogram

**Example:**

HistogramScale = (0,1000);

**Attributes**

HistogramMinValue  HistogramMaxValue

**HistogramMinValue**

Number specifying the lower value for the data range of the histogram. Data points below the minimum will be placed in the first bin.

**HistogramMaxValue**

Number specifying the upper value for the data range of the histogram. Data points above the maximum will be placed in the last bin.

**HistogramType**

HistogramType = HistogramType;

HistogramType describes the type of data distribution in a histogram.
**Common CDL Attributes**

**Used in These Charts**
Histogram

**Example:**
```plaintext
HistogramType = BYNUMBER;
```

**Attributes**
*HistogramType*

**HistogramType**

String constant describing the distribution type. Legal values are BYNUMBER, PERCENTAGE, and PROBABILITY. The default is BYNUMBER.

- **BYNUMBER** specifies that a bin will hold a count of the number of data points in the bin's range.
- **PERCENTAGE** specifies that a bin will hold the percentage of the total data points that are in the bin's range.
- **PROBABILITY** specifies that a bin will hold the probability that any given point is in the bin's range.

**LabelAnimationStyle**

LabelAnimationStyle = FADE | NONE

Defines how axis and data labels initially appear in a chart. This parameter is only valid in SVG and SVGweb output formats.

**Example:**
```plaintext
LabelAnimationStyle = FADE;
```

**Values**

- **FADE** The labels fade in.
- **NONE** The labels are immediately visible.

**Default**
NONE
Labels

TopLabels[N] = "Label1", "Label2", ...;
BottomLabels[N] = "Label1", "Label2", ...;
LeftLabels[N] = "Label1", "Label2", ...;
RightLabels[N] = "Label1", "Label2", ...;

The Labels parameter specifies a list of custom tic mark labels that will be used instead of the numeric labels automatically generated by the axis. The Labels will be evenly placed along the axis, overriding any tic placement specified by the StepValue attribute. If a corresponding TicLocations parameter is specified, then the labels will be drawn in order at the locations specified in TicLocations[N].

In a Bar, Combo, Pareto or Stock Chart, the BarLabels parameter overrides the LeftLabels (for horizontal bars) parameters.

Used in These Charts
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

Example:

Attributes
Label

LabelPos

LabelPos = Float;

Defines the position of a pie slice label relative to the width of the pie. A value of 1.1 will place each label just outside the pie, while a value of 0.6 will place each label inside of each pie slice.

Used in These Charts
Pie

Example:
LabelPos = 1.1;

Attributes
Float

Float

Float is a number indicating slice label placement, where 1.0 is on the outside edge of the pie.
Common CDL Attributes

Example:

```
LabelPos = 1.1;
LabelPos = 0.6;
```

Values

< 1.0  Places the labels inside the pie border.
> 1.0  Places the labels outside the pie border

Default

None

Layout

```
Layout = (LayoutType);
```

The Layout parameter is used to specify the layout format for all of the charts defined in the Charts parameter. The examples below show Layout used in context.

Used in These Charts

Multi-chart

Example:

```
Layout = ROWS;
Charts = ("Sales", PIECHART),
       ("Expense", BARCHART),
       ("Bonus", XYCHART),
       (BREAK),
       ("Growth", COMBOCHART),
       (BREAK),
       ("Budget", XYCHART),
       ("Salary", PIECHART);

<PARAM NAME=Layout VALUE="(ROWS)">
<PARAM NAME = ChartScript1 VALUE='
Background = (white, NONE);
Header = ("1. Most Requested Pages", black, Helvetica, 12);
DwellLabel = (",", black, "Helvetica", 9);
DwellBox = (xe3e3e3, SHADOW, 2);
ColorTable = xB5D5F0, xBEA9AD, xDACE98, xEBF0F3, xAABAC5, xBFC1A0;
BottomTics = ("ON", black, "Helvetica", 9);
LeftTics = ("ON", black, "Helvetica", 9);
LeftScale = (0, 2700);
LeftFormat = (INTEGER);
BarLabels = "Home\nPage", "NetCharts", "Examples", "Products";
GraphType = GROUP;
DataSets = ("Server1", NULL);
DataSet1 = 2694, 780, 628, 513;
Bar3DDepth = 3;
'>
```
Attributes

**LayoutType**

*LayoutType* can be either **ROWS** or **COLS**, specifying that the charts should be arranged in rows or columns, respectively. The default layout is **COLS**.

**Example:**

```plaintext
Layout = ROWS;                  <!-- charts laid out in rows -->
Charts = ("Sales", PIECHART),
        ("Expense", BARCHART),
        ("Bonus", XYCHART),
        (BREAK),
        ("Growth", COMBOCHART),
        (BREAK),
        ("Budget", XYCHART),
        ("Salary", PIECHART);

<PARAM NAME=Layout VALUE="(ROWS)">
<PARAM NAME = ChartScript1 VALUE='
Background = (white, NONE);
Header = ("1. Most Requested Pages", black, Helvetica, 12);
DwellLabel = ("", black, "Helvetica", 9);
DwellBox = (xe3e3e3, SHADOW, 2);
ColorTable = xB5D5F0, xBEA9AD, xDACE98, xEBF0F3, xAABAC5, xBFC1A0;
BottomTics = ("ON", black, "Helvetica", 9);
LeftTics = ("ON", black, "Helvetica", 9);
LeftScale = (0, 2700);
LeftFormat = (INTEGER);
BarLabels = "Home\nPage", "NetCharts", "Examples", "Products";
GraphType = GROUP;
DataSets = ("Server1", NULL);
DataSet1 = 2694, 780, 628, 513;
Bar3DDepth = 3;
'>
```

**Values**

<table>
<thead>
<tr>
<th>LayoutType</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROWS</td>
<td>Charts will be arranged in rows</td>
</tr>
<tr>
<td>COLS</td>
<td>Charts will be arranged in columns</td>
</tr>
</tbody>
</table>

**Default**

COLS

---

**LeftActiveLabels**

`LeftActiveLabels = ("Label", "URL", "Target"), ...;`

The left axis labels become active labels when *LeftActiveLabels* is used. Each set in parenthesis has a corresponding set within a *DataSet* parameter.
**Used in These Charts**

All

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
</table>

**LeftAxis**

LeftAxis = (Label, Color, FontName, FontSize, Angle, interiorAlignment);

If LeftAxis is defined for a Combo chart, then the top axis will be used to map the X data values for all line sets, unless otherwise specified using the LineAxis parameter. The group sets the typographic characteristics for the data values.

| interiorAlignment | Specifies the alignment to use in text strings that contain multiple lines. |

The legal values for interiorAlignment are LEFT, RIGHT, or CENTER.

**Used in These Charts**

Combo

**Example:**

```
LeftAxis = ("", black, "TimesRoman", 16, 0);
```

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>Color</th>
<th>FontName</th>
<th>FontSize</th>
<th>Angle</th>
<th>interiorAlignment</th>
</tr>
</thead>
</table>

**LeftAxisTitle**

LeftAxisTitle = (Label, Color, FontName, FontSize, Angle, interiorAlignment, exteriorAlignment);

The LeftAxisTitle parameter specifies the label attributes for the axis title, which centered along the top axis, just above the grid. When the Header parameter, whether because of the use of a legend, or for some other reason, creates a title that seems visually unbalanced, you may find this parameter produces a more pleasing chart title.

| interiorAlignment | Specifies the alignment to use in text strings that contain multiple lines. |
| exteriorAlignment | Specifies the alignment for the entire Title object. |

The legal values for interiorAlignment and exteriorAlignment are LEFT, RIGHT, or CENTER.
Common CDL Attributes

Used in These Charts
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

Example:
LeftAxisTitle = ("Ceres Prototype Project Schedule\n", black, "Helvetica", 12);

Attributes
Label  Color  FontName  FontSize  Angle  interiorAlignment  exteriorAlignment

LeftAxisTitleActiveLabel

LeftAxisTitleActiveLabel = ("Label", "URL", "Target");

LeftAxisTitleActiveLabel defines a single active label destination for the LeftAxisTitle parameter.

Used in These Charts
All

Example:
LeftAxisTitleActiveLabel = ("Destination", "demo.html", "framelm");

Attributes
Label  URL  Target

LeftAxisTitleBox

LeftAxisTitleBox = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor,
TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);

The LeftAxisTitleBox parameter specifies the region attributes for the axis title centered along the axis. The image-related attributes need not be used unless you want to use an image texture on the box.

Used in These Charts
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

Example:
LeftAxisTitleBox = (lightgray, SHADOW, 3,,gray);

Attributes
Color  BorderType  BorderWidth  ImageURL  ImageFormat  BorderColor  TRCornerStyle  BRCornerStyle  BLCornerStyle  CornerColor
**XXCornerStyle**

The drawing style for each of the four corners of the region. Styles are specified in a clockwise fashion starting in the upper left - Top Left, Top Right, Bottom Right, and Bottom Left. Legal values are SQUARE, SNIP and ROUND. The default is SQUARE.

**LeftColor**

```
LeftColor = Color;
```

`LeftColor` controls the color of the top axis and the tic marks, but not the tic mark labels. The default axis color is black. If the NULL color is specified, the axis color is not changed by this parameter.

**Example:**

```
LeftAxisColor = xB5D5F0;
```

**Attributes**

*Color*

**LeftDrawMinorTics**

```
LeftDrawMinorTics = ON|OFF;
```

`LeftDrawMinorTics` controls whether or not left tics are drawn. The default value is ON.

**Example:**

```
LeftDrawMinorTics = OFF;
```

**Attributes**

*Switch*

**LeftFormat**

```
LeftFormat = (FormatType, "FormatExpr", "TimeBase", "TimeUnit");
```

`LeftFormat` adjusts the numeric labels that are automatically generated for the top axis, should one be defined.

**Used in These Charts**

Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y
Example:

```java
LeftFormat = (DATE, "%w\n%M/%D", "1 Jan 2000", "1h");
LeftFormat = (DATE, "%M/%D", "1 Apr 96", "1d");
LeftFormat = (INTEGER);
LeftFormat = (FLOAT,"$%,9.2f",,);
```

Attributes

<table>
<thead>
<tr>
<th>FormatType</th>
<th>FormatExpr</th>
<th>TimeBase</th>
<th>TimeUnit</th>
</tr>
</thead>
</table>

### FormatType

FormatType specifies the type of number being displayed on the top axis.

#### Example:

```java
LeftFormat = (DATE, "%w\n%M/%D", "1 Jan 2000", "1h");
LeftFormat = (DATE, "%M/%D", "1 Apr 96", "1d");
LeftFormat = (INTEGER);
LeftFormat = (FLOAT,"$%,9.2f",,);
```

### Values

<table>
<thead>
<tr>
<th>DATE</th>
<th>Axis value are shown as date and/or time values. See Appendix A: Date and Time Values for further detail.</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLOAT</td>
<td>Axis values are shown with decimal parts.</td>
</tr>
<tr>
<td>INTEGER</td>
<td>Axis values are shown as integers, and are rounded if necessary.</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>Axis values are shown as decimals. See <a href="http://java.sun.com/j2se/1.3/docs/api/java/text/DecimalFormat.html">http://java.sun.com/j2se/1.3/docs/api/java/text/DecimalFormat.html</a> for more information.</td>
</tr>
</tbody>
</table>

The characters listed here are used in non-localized patterns. Localized patterns use the corresponding characters taken from this formatter's DecimalFormatSymbols object instead, and these characters lose their special status. Two exceptions are the currency sign and quote, which are not localized.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Location</th>
<th>Localized?</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Number</td>
<td>Y</td>
<td>Digit</td>
</tr>
<tr>
<td>#</td>
<td>Number</td>
<td>Y</td>
<td>Digit, zero shows as absent</td>
</tr>
<tr>
<td>.</td>
<td>Number</td>
<td>Y</td>
<td>Decimal separator or monetary decimal separator</td>
</tr>
<tr>
<td>-</td>
<td>Number</td>
<td>Y</td>
<td>Minus sign</td>
</tr>
<tr>
<td>,</td>
<td>Number</td>
<td>Y</td>
<td>Grouping separator</td>
</tr>
<tr>
<td>E</td>
<td>Number</td>
<td>Y</td>
<td>Separates mantissa and exponent in scientific notation. <em>Need not be quoted in prefix or suffix.</em></td>
</tr>
<tr>
<td>;</td>
<td>Subpattern boundary</td>
<td>Y</td>
<td>Separates positive and negative subpatterns</td>
</tr>
<tr>
<td>%</td>
<td>Prefix or suffix</td>
<td>Y</td>
<td>Multiply by 100 and show as percentage</td>
</tr>
<tr>
<td>\u2030</td>
<td>Prefix or suffix</td>
<td>Y</td>
<td>Multiply by 1000 and show as per mille</td>
</tr>
<tr>
<td>\ı (\u00A4)</td>
<td>Prefix or suffix</td>
<td>N</td>
<td>Currency sign, replaced by currency symbol. If doubled, replaced by international currency symbol. If present in a pattern, the</td>
</tr>
</tbody>
</table>
### Common CDL Attributes

<table>
<thead>
<tr>
<th>Prefix or suffix</th>
<th>N</th>
<th>monetary decimal separator is used instead of the decimal separator.</th>
</tr>
</thead>
</table>

**Default**

INTEGER

**TimeBase**

The **TimeBase** attribute specifies the base date to be used when determining the actual date or time value when using a time unit or numeric value. See **Appendix A: Date and Time** Values for further detail.

**Example:**

LeftFormat = (DATE, "%w\n%M/%D", "1 Jan 2000", "1h");
LeftFormat = (DATE, "%M/%D", "1 Apr 96", "1d");

**Values**

String values representing dates or times

**Default**

None

**TimeUnit**

The **TimeUnit** attribute controls the time multiplier to be used when determining the actual date/time value when using a numeric value. See **Appendix A: Date and Time** Values for further detail.

**Example:**

LeftFormat = (DATE, "%w\n%M/%D", "1 Jan 2000", "1h");
LeftFormat = (DATE, "%M/%D", "1 Apr 96", "1d");

**Values**

String values representing dates or times

**Default**

None
**LeftLabels**

LeftLabels = "Label1", "Label2", ...;

The `LeftLabels` parameter specifies a list of custom tic mark labels that will be used instead of the numeric labels automatically generated by the axis. The `LeftLabels` will be evenly placed along the axis, overriding any tic placement specified by the `StepValue` attribute.

In a Bar, Combo, Pareto, or Stock Chart, the `BarLabels` parameter overrides the `LeftLabels` (for horizontal bars) parameters.

**Used in These Charts**

Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**


**Attributes**

Label

---

**LeftMargins**

LeftMargins = (BottomSideMargin, TopSideMargin);

The `LeftMargins` parameter specifies the gap, in pixels, at the beginning and end of the left axis. Most often used to prevent clipping of data points at the extreme ends of the scale.

**Example:**

LeftMargins = (20, 20);

**Attributes**

Label

---

**LeftScroll**

LeftScroll = (ScrollMin, ScrollMax);

The `LeftScroll` parameter specifies a range of values through which the bottom axis can be scrolled. When the `ScrollMin` and `ScrollMax` attributes are defined for the axis, the axis will be displayed as a slider bar, using the axis color defined, with a white background. The relative size of the slider represents the percentage of the entire range currently being displayed. That is, it graphically depicts the size of the current axis range (MinValue and MaxValue attributes) relative to the scrollable region (ScrollMin and ScrollMax attributes). See the `LeftScale` parameter for MinValue and MaxValue definitions.

`LeftScroll` should only be used in conjunction with the `LeftScale` parameter.

**Example:**

LeftScroll = (0, 98);

**Attributes**

ScrollMin ScrollMax
**ScrollMin**

ScrollMin sets the lower visible limit for a scrollbar defined with LeftScroll

**Example:**
```
TopScroll = (0, 98);
```

**Values**

<MinValue

**Default**

None

**ScrollMax**

ScrollMax sets the upper visible limit for a scrollbar defined with BottomScroll

**Example:**
```
LeftScroll = (0, 98);
```

**Values**

> MaxValue

**Default**

None

**LeftTicLength**

```
LeftTicLength = Number;
```

The LeftTicLength parameter defines the size of axis tic marks which are displayed along the left axis of a chart. The parameter will reset the automatically generated tic length. The value defines the number of pixels to use for the length of the tic mark. By default, the number of pixels used is the width of the character zero (0) as found in the font applied to the label. Setting the LeftTicLength to the value -1 will cause the default size to be used.

**Attributes**

Number

**Number**

Apparent length of a left axis tic mark in a chart, in pixels.
**Common CDL Attributes**

**Number**

**Used in These Charts**
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**

```
LeftTicLength = 10;
```

**Values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No effect (zero length tics are not drawn).</td>
</tr>
<tr>
<td>1 or greater</td>
<td>Whole number length in pixels</td>
</tr>
</tbody>
</table>

**Default**

-1

**LeftTitle**

```
LeftTitle = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment, exteriorAlignment);
```

**LeftTitle** describes an optional title, or label, that sits on the left side of a chart, and uses standard attributes for string text, text color, font, font size, and label rotation. As with Header, **LeftTitle** is universally available in Visual Mining chart applications.

**Used in These Charts**
All, but most useful in Pie, Diagram, and Dial, which do not have axes.

**Example:**

```
LeftTitle = ("Financial Status", royalblue, Helvetica, 14, 0, LEFT);
```

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interiorAlignment</td>
<td>Specifies the alignment to use in text strings that contain multiple lines.</td>
</tr>
<tr>
<td>exteriorAlignment</td>
<td>Specifies the alignment for the entire Title object.</td>
</tr>
<tr>
<td>extend</td>
<td>The specified background should extend entire length of chart.</td>
</tr>
</tbody>
</table>

The legal values for **interiorAlignment** and **exteriorAlignment** are LEFT, RIGHT, or CENTER. The legal values for **extend** are ON and OFF.

**Attributes**

Label, Color, FontName, FontSize, Angle, interiorAlignment, exteriorAlignment

**LeftTitleActiveLabel**

```
LeftTitleActiveLabel = ("Label", "URL", "Target");
```

**LeftTitleActiveLabel** defines a single active label destination for the **LeftTitle** parameter.
Used in These Charts
All

Example:
    LeftActiveLabel = ("Destination", "demo.html", "frame1");

Attributes
Label          URL          Target

LeftTitleBox

LeftTitleBox = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);

The LeftTitleBox specifies a background region just for the LeftTitle parameter.

Used in These Charts
All

Attributes
BorderColor      BorderType      BorderWidth      Color
ImageFormat      ImageURL        TRCornerStyle    BRCornerStyle
BLCornerStyle    CornerColor

XXCornerStyle

The drawing style for each of the four corners of the region. Styles are specified in a clockwise fashion starting in the upper left - Top Left, Top Right, Bottom Right, and Bottom Left. Legal values are SQUARE, SNIP and ROUND. The default is SQUARE.

LeftTics

LeftTics = ("Mode", Color, "FontName", FontSize, Angle, interiorAlignment);

The LeftTics parameter specifies the label attributes for the tic marks displayed along the left axis. The tic labels are generated automatically based on the other parameter settings, and are displayed using the given label attributes in the LeftTics parameter. If any attribute is not defined, any previous value of that attribute will be used.

<table>
<thead>
<tr>
<th>interiorAlignment</th>
<th>Specifies the alignment to use in text strings that contain multiple lines.</th>
</tr>
</thead>
<tbody>
<tr>
<td>backgroundColor</td>
<td>Background color of the tic label area</td>
</tr>
<tr>
<td>rotationPoint</td>
<td>For rotated axis labels, anchor point for the rotation</td>
</tr>
</tbody>
</table>
The legal values for `interiorAlignment` are **LEFT**, **RIGHT**, or **CENTER**.

The legal values for `rotationPoint` are **LEFT**, **RIGHT**.

**Used in These Charts**
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**

```plaintext
LeftTics = ("OFF", black, "Helvetica", 10,LEFT,,RIGHT);
```

**Attributes**

<table>
<thead>
<tr>
<th>Mode</th>
<th>Color</th>
<th>FontName</th>
<th>FontSize</th>
<th>Angle</th>
<th>interiorAlignment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>InteriorAlignment</strong></td>
<td><strong>BackgroundColor</strong></td>
<td><strong>RotationPoint</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Mode**

Mode determines whether or not the tic labels are shown on that axis.

**Example:**

```plaintext
LeftTics = ("OFF", black, "Helvetica", 10,LEFT,,RIGHT);
```

**Values**
ON Show tic labels for this axis
OFF Don’t show tic labels on this axis

**Default**
ON

**Legend**

Legend = ("Label", Color, "FontName", FontSize, Angle, interiorAlignment);

All Visual Mining charts, except for the Diagram chart and Multi-Chart, support the display of a legend anywhere on the chart. Legends offer a key to the data sets. The entries for the `Legend` parameter can be fully specified, or can be loaded automatically using the names and colors assigned to all data sets being displayed.

<table>
<thead>
<tr>
<th>interiorAlignment</th>
<th>Specifies the alignment to use in text strings that contain multiple lines.</th>
</tr>
</thead>
</table>

The legal values for `interiorAlignment` are **LEFT**, **RIGHT**, or **CENTER**.
Common CDL Attributes

Used in These Charts
Box, Bar, Bubble, Combo, Dial, Line, Pareto, Pie, Radar, Stock, Strip, Time, X-Y

Example:
Legend = ("Last Quarter", blue, Helvetica, 12, 270);

Attributes
Label    Color    FontName    FontSize    Angle    interiorAlignment

LegendActiveLabels
LegendActiveLabels = ("Label", "URL", "Target"), ...;

LegendActiveLabels define sets of active label destinations for the Legend parameter.

Used in These Charts
All

Example:
LegendActiveLabels = ("Set 1", "demo1.html", "frame1"),
("Set 2", "demo2.html", "frame1"),
("Set 3", "demo3.html", "frame1");

Attributes
Label    URL    Target

LegendAnimationStyle
LegendAnimationStyle = FADE | NONE

Defines how the legend initially appears in a chart. This parameter is only valid in SVG or SVGWeb output formats.

Example:
LegendAnimationStyle = FADE;

Values
FADE          The legend fades in.
NONE          The legend is immediately visible.

Default
NONE
**LegendAxis**

\[
\text{LegendAxis} = (\text{XAxis}, \text{Yaxis});
\]

The `LegendAxis` parameter is optional and is only used if the `LegendLayout` location parameter is `CENTER` or the `LegendBoxSize` parameter is used. This parameter defines the types of coordinates used for the X and Y attributes in the `LegendLayout` parameter and for the `MaxWidth` and `MaxHeight` attributes in the `LegendBoxSize` parameter.

**Used in These Charts**

All

**Example:**

\[
\text{LegendAxis} = (\text{Bottom}, \text{Right});
\]

**Attributes**

<table>
<thead>
<tr>
<th>XAxis</th>
<th>Yaxis</th>
</tr>
</thead>
</table>

**XAxis**

Specifies which side of a Legend becomes the X axis.

**Example:**

\[
\text{LegendAxis} = (\text{Bottom}, \text{Right});
\]

**Values**

<table>
<thead>
<tr>
<th>Percent</th>
<th>Values will be between 0 and 1.0 or 0 and 100, being the total width or height of the graph.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pixel</td>
<td>Raw pixel coordinates</td>
</tr>
<tr>
<td>BOTTOM</td>
<td>Map values with the bottom axis of the chart</td>
</tr>
<tr>
<td>TOP</td>
<td>Map values with the top axis of the chart</td>
</tr>
</tbody>
</table>

**Default**

PERCENT

**YAxis**

Specifies which side of a Legend becomes the Y axis.

**Example:**

\[
\text{LegendAxis} = (\text{Bottom}, \text{Right});
\]
Values
Percent  Values will be between 0 and 1.0 or 0 and 100, being the total width or height of the graph.
Pixel     Raw pixel coordinates
LEFT     Map values with the left side of the chart
RIGHT    Map values with the right side of the chart

Default
PERCENT

LegendBox

LegendBox = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor);

The LegendBox specifies a background region just for the Legend parameter, and is optional, but may only appear with the Legend parameter. If specified, it defines a box to be displayed as a background for the Legend. The box will be automatically scaled to fit all the legend items, unless otherwise restricted by the LegendBoxSize parameter.

Used in These Charts
All

Example:
LegendBox = (yellow, SHADOW, 5);

Attributes
BorderColor  BorderType  BorderWidth  Color  ImageFormat
ImageURL

LegendBoxSize

LegendBoxSize = (MaxWidth, MaxHeight);

The LegendBoxSize parameter is optional, but may only appear with the Legend parameter. LegendBoxSize is used to specify maximum size values for the legend. This is useful in cases where the legend text must be automatically clipped.

Used in These Charts
All

Example:
LegendBoxSize = (100, 150);
**MaxHeight**

MaxHeight only limits the height of the legend if it grows too large; it does not explicitly set the height of the legend. If specified, this attribute is also used when automatically determining the number of columns for the legend.

**Example:**

```
LegendBoxSize = (100, 150);
```

**Values**

Maximum allowable legend box height in pixels

**Default**

0

**MaxWidth**

MaxWidth only limits the width of the legend if it grows too large; it does not explicitly set the width of the legend. If specified, this attribute is also used when automatically determining the number of columns for the legend.

**Example:**

```
LegendBoxSize = (100, 150);
```

**Values**

Maximum allowable legend box width in pixels

**Default**

0

**LegendDwellAnimationHighlightBorderStyle**

LegendDwellAnimationHighlightBorderStyle = (lineType, LineWidth, lineColor);

Defines the border style to be applied to data points in a series when LegendDwellAnimationStyle = HIGHLIGHT.

**Used in These Charts**

All
**Example:**

```
LegendDwellAnimationHighlightBorderStyle = (SOLID,1,BLACK);
```

### Attributes

<table>
<thead>
<tr>
<th>LineType</th>
<th>LineWidth</th>
<th>LineColor</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Values</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SOLID</td>
<td>A solid line is displayed.</td>
<td></td>
</tr>
<tr>
<td>DOTTED</td>
<td>A dotted line is displayed.</td>
<td></td>
</tr>
<tr>
<td>DASHED</td>
<td>A dashed line is displayed.</td>
<td></td>
</tr>
<tr>
<td>DOTDASH</td>
<td>Alternating dots and dashes are displayed.</td>
<td></td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>SOLID</td>
<td></td>
</tr>
</tbody>
</table>

### LegendDwellAnimationHighlightFill

```
LegendDwellAnimationHighlightFill = Color;
```

Defines the color used to fill data points in a series when `LegendDwellAnimationStyle = HIGHLIGHT`.

### Used in These Charts

All

### Example:

```
LegendDwellAnimationHighlightFill = BLUE);
```
Common CDL Attributes

LineColor

Attributes
Color

Default
NONE

LegendDwellAnimationStyle

LegendDwellAnimationStyle = HIGHLIGHT | NONE

Defines how the chart behaves when the mouse dwells over a legend entry.

Example:
LegendDwellAnimationStyle = HIGHLIGHT;

Values
HIGHLIGHT All data points in the related series are highlighted using values specified in
LegendDwellAnimationHighlightFill and
LegendDwellAnimationHighlightBorderStyle.
NONE No highlight is applied to the data points.

Default
NONE

LegendItems

LegendItems = ("Label1", Color1, SymType1, SymSize1, SymStyle1, LineType1, LineWidth1, LineColor1,
PatternFill, color1, color2, imageURL, shadowwidth), ...;

The LegendItems parameter is optional, but may only appear with the Legend parameter. If specified,
it defines one or more items to be included in the Legend, each with its own attributes within the
parenthesized set.

Used in These Charts

All
### Example:
```csharp
LegendItems = ("1st Data Set", peachpuff, DIAMOND, 5, FILLED, SOLID, 2, peachpuff),
("2nd Data Set", mintcream, DIAMOND, 5, FILLED, SOLID, 2, mintcream),
("3rd Data Set", plum, DIAMOND, 5, FILLED, SOLID, 2, plum);
```

### Attributes

<table>
<thead>
<tr>
<th>Label</th>
<th>Color</th>
<th>SymType</th>
<th>SymSize</th>
</tr>
</thead>
<tbody>
<tr>
<td>SymStyle</td>
<td>LineType</td>
<td>LineWidth</td>
<td>LineColor</td>
</tr>
</tbody>
</table>

#### SymSize

SymSize indicates the size, in pixels, of the iconic symbol to use for this legend item.

**Example:**
```csharp
LegendItems = ("1st Data Set", peachpuff, DIAMOND, 5, FILLED, SOLID, 2, peachpuff),
```

**Values**

- Integer values of pixels

**Default**

0

#### SymStyle

SymStyle indicates the style, either filled or outlined, of the iconic symbol to use for this legend item.

**Example:**
```csharp
LegendItems = ("1st Data Set", peachpuff, DIAMOND, 5, FILLED, SOLID, 2, peachpuff),
```

**Values**

- FILLED
- OUTLINED

**Default**

FILLED

#### SymType

SymType indicates the kind of iconic symbol to use for this legend item.

**Example:**
```csharp
LegendItems = ("1st Data Set", peachpuff, DIAMOND, 5, FILLED, SOLID, 2, peachpuff),
```
### Common CDL Attributes

#### PatternFill

**Values**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>No symbol is displayed.</td>
</tr>
<tr>
<td>CIRCLE</td>
<td>Displays circles</td>
</tr>
<tr>
<td>SQUARE</td>
<td>Displays squares</td>
</tr>
<tr>
<td>DIAMOND</td>
<td>Displays diamonds</td>
</tr>
<tr>
<td>CROSS</td>
<td>Displays crosses</td>
</tr>
<tr>
<td>TARGET</td>
<td>Displays targets (bulls-eye)</td>
</tr>
<tr>
<td>TRIANGLEDOWN</td>
<td>Displays downward pointing triangles</td>
</tr>
<tr>
<td>TRIANGLEUP</td>
<td>Displays upward pointing triangles</td>
</tr>
</tbody>
</table>

**Default**

SQUARE

**PatternFill**

The `PatternFill` attribute provides a visual pattern fill for a legend item. The `Color1` and `Color2` attributes provide colors to use in creating the pattern fill.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>No pattern, do default fill, if any</td>
</tr>
<tr>
<td>Built-In Patterns</td>
<td></td>
</tr>
<tr>
<td>FSLASH</td>
<td>Front slash type</td>
</tr>
<tr>
<td>BSLASH</td>
<td>Back slash type</td>
</tr>
<tr>
<td>DGRID</td>
<td>Diagonal grid lines, (front and back slash lines)</td>
</tr>
<tr>
<td>HORIZONTAL</td>
<td>Horizontal lines</td>
</tr>
<tr>
<td>VERTICAL</td>
<td>Vertical lines</td>
</tr>
<tr>
<td>GRID</td>
<td>Grid lines, (horizontal and vertical lines)</td>
</tr>
<tr>
<td>Gradients</td>
<td></td>
</tr>
<tr>
<td>GRADIENTVERTICAL</td>
<td>Bottom to top style gradient</td>
</tr>
<tr>
<td>GRADIENTHORIZONTAL</td>
<td>Left to right style gradient</td>
</tr>
<tr>
<td>GRADIENTFDIAG</td>
<td>Top right to bottom left style gradient</td>
</tr>
<tr>
<td>GRADIENTBDIAG</td>
<td>Top left to bottom right style gradient</td>
</tr>
<tr>
<td>GRADIENTRADIAL</td>
<td>Radial style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERHORIZONTAL</td>
<td>Center out horizontal style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERVERTICAL</td>
<td>Center out vertical style gradient</td>
</tr>
<tr>
<td>Images</td>
<td></td>
</tr>
<tr>
<td>IMAGE</td>
<td>Use an image specified in the optional imageURL element</td>
</tr>
</tbody>
</table>

**color1**

This color is used in the following ways:
- Foreground color for patterns
- Starting color for gradients
- Ignored in images

**color 2**

This color is used in the following ways:
- Background color for patterns
- Stopping color for gradients
LegendLayout

LegendLayout = (Type, Location, X, Y, Justify, Columns);

The LegendLayout parameter is optional, but may only appear with the Legend parameter. If specified, it defines the placement of the legend within the chart.

Used in These Charts
All

Example:
LegendLayout = (HORIZONTAL, BOTTOM, 0, 0, LEFT, 2);

Attributes

<table>
<thead>
<tr>
<th>Columns</th>
<th>Justify</th>
<th>Location</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Columns

The Type and Columns attributes specify the way legend items are displayed within the legend box. In general, the Columns attribute specifies the desired number of text columns.

Example:
LegendLayout = (HORIZONTAL, BOTTOM, 0, 0, LEFT, 2);

Values

< 0  The number of columns will be automatically determined in order to prevent the legend height from exceeding the height of the graph or the maximum height specified by the LegendBoxSize parameter.
0 or 1 A single column will be displayed.
> 1  The specified number of columns will be displayed.

Default
-1

Justify

When using a Location value of CENTER, the Justify attribute determines where the legend box is located relative to the legend control point specified by X and Y, as shown below:

| TOPLEFT | TOP | TOPRIGHT |

LegendBoxSize

The LegendBoxSize parameter allows the specified area to be divided evenly. The number of columns is determined by the number of rows in the chart. If the number of columns is less than the specified value, then the table is displayed in the row with the largest number of columns. If the number of columns is greater than the specified value, then the table is displayed in as many rows as necessary.
For example, if Justify is set to TOP, then the legend will be displayed below the control point. That is, the top of the legend will be positioned at the control point. If, on the other hand, BOTTOMRIGHT is chosen, then the bottom right corner of the legend will be positioned at the control point.

Example:

```
LegendLayout = (HORIZONTAL, BOTTOM, 100, 100, BOTTOMRIGHT, 2);
```

Values

<table>
<thead>
<tr>
<th>Location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOPLEFT</td>
<td>Displayed at the top of the panel and is left justified</td>
</tr>
<tr>
<td>TOP</td>
<td>Displayed at the top of the panel and is center justified</td>
</tr>
<tr>
<td>TOPRIGHT</td>
<td>Displayed at the top of the panel and is right justified</td>
</tr>
<tr>
<td>LEFT</td>
<td>Displayed at the left of the panel and is left justified</td>
</tr>
<tr>
<td>CENTER</td>
<td>X and Y attributes used to specify legend control point.</td>
</tr>
<tr>
<td>RIGHT</td>
<td>Displayed at the right of the panel and is right justified</td>
</tr>
<tr>
<td>BOTTOMLEFT</td>
<td>Displayed at the bottom of the panel and is left justified</td>
</tr>
<tr>
<td>BOTTOM</td>
<td>Displayed at the bottom of the panel and is center justified</td>
</tr>
<tr>
<td>BOTTOMRIGHT</td>
<td>Displayed at the bottom of the panel and is right justified</td>
</tr>
</tbody>
</table>

Default

CENTER

Location

The Location attribute defines the location of the legend relative to the graph. All of the location values, except for CENTER, specify fixed areas outside of the graph, and the size of the legend will affect the size of the graph. In those cases, the exact placement of the legend is completely controlled by the chart, with appropriate justification. For example, a TOP chart legend is displayed at the top of the panel and is center justified, while a BOTTOMRIGHT legend is displayed below the graph and is right justified.

If the Location is set to CENTER, then the X, Y attributes are used to specify the location of the legend control point. The control point can be specified using a number of different coordinate types (see LegendAxis parameter for details) but in all cases, it represents an anchor position within the confines of the chart itself. That is, the control point is a location that lies inside of the axes. As such, any legend displayed using the CENTER location will be clipped to the borders of the chart.

When using the CENTER location, the Justify attribute determines where the legend box is located relative to the legend control point specified by X and Y.

Example:

```
LegendLayout = (HORIZONTAL, BOTTOM, 100, 100, BOTTOMRIGHT, 2);
```
### Common CDL Attributes

#### Type

<table>
<thead>
<tr>
<th>Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOPLEFT</td>
<td>Displayed at the top of the panel and is left justified</td>
</tr>
<tr>
<td>TOP</td>
<td>Displayed at the top of the panel and is center justified</td>
</tr>
<tr>
<td>TOPRIGHT</td>
<td>Displayed at the top of the panel and is right justified</td>
</tr>
<tr>
<td>LEFT</td>
<td>Displayed at the left of the panel and is left justified</td>
</tr>
<tr>
<td>CENTER</td>
<td>X and Y attributes used to specify legend control point</td>
</tr>
<tr>
<td>RIGHT</td>
<td>Displayed at the right of the panel and is right justified</td>
</tr>
<tr>
<td>BOTTOMLEFT</td>
<td>Displayed at the bottom of the panel and is left justified</td>
</tr>
<tr>
<td>BOTTOM</td>
<td>Displayed at the bottom of the panel and is center justified</td>
</tr>
<tr>
<td>BOTTOMRIGHT</td>
<td>Displayed at the bottom of the panel and is right justified</td>
</tr>
</tbody>
</table>

#### Default

**RIGHT**

**Type**

The **Type** and **Columns** attributes specify the way legend items are displayed within the legend box. In general, the **Type** attribute specifies the desired orientation and the **Columns** attribute specifies the desired number of columns.

**Example:**

```
LegendLayout = (HORIZONTAL, BOTTOM, 100, 100, BOTTOMRIGHT, 2);
```

#### Values

- **HORIZONTAL**
  
The number of columns will be automatically determined in order to prevent the legend width from exceeding the width of the graph or the maximum width specified by the **LegendBoxSize** parameter.
  
  If **Columns** is 0 or 1, a single column will be displayed.

- **VERTICAL**
  
The number of columns will be automatically determined in order to prevent the legend height from exceeding the height of the graph or the maximum height specified by the **LegendBoxSize** parameter.
  
  If **Columns** is 0 or 1, a single row will be displayed.

#### Default

**VERTICAL**

**X**

Specifies the X position of the control point of the Legend.

**Example:**

```
LegendLayout = (HORIZONTAL, BOTTOM, 100, 100, BOTTOMRIGHT, 2);
```
Common CDL Attributes

Values
Percent  Values will be between 0 and 1.0 or 0 and 100, being the total width or height of the graph.
Pixel     Raw pixel coordinates
BOTTOM   Map values with the bottom axis of the chart
TOP      Map values with the top axis of the chart

Default
50

Y

Specifies the Y position of the control point of the Legend

Example:
LegendLayout = (HORIZONTAL, BOTTOM, 100, 100, BOTTOMRIGHT, 2);

Values
Percent  Values will be between 0 and 1.0 or 0 and 100, being the total width or height of the graph.
Pixel    Raw pixel coordinates
LEFT     Map values with the left side of the chart
RIGHT    Map values with the right side of the chart

Default
50

Line3Ddepth

Line3Ddepth[N] = depth;

If Line3DDepth is specified, then the lines drawn on the chart can be “3D.” This parameter can be used with the LineWidth parameter to achieve visible separation between 3D line sets.

Used in These Charts
Combo, Line, Pareto, Stock, X-Y

Example:
Line3DDepth = 17;

Attributes
Depth

Depth

Depth indicates the width of the line along a Z-axis.
**Example:**

```
Line3DDepth = 17;
```

**Values**

- **>1** If depth is greater than 1, then all line sets will be displayed as 3D lines. The max amount of space allocated to each line in the Z dimension will be "depth" pixels.
- **1** Lines will not be seen as 3D

**Default**

1

---

**LineAnimationStyle**

```
LineAnimationStyle = BEND | FADE | NONE
```

Specifies how lines initially appear in a line chart. This parameter is only valid in SVG or SVGWeb output formats.

**Attributes**

- **Style**

**Style**

Style refers to the manner in which lines are first rendered in a line chart.

**Example:**

```
LineAnimationStyle = BEND;
```

**Values**

- **BEND** The lines start as a straight line at zero and each point bends to its actual value.
- **FADE** The lines fade in.
- **NONE** The lines are immediately visible.

**Default**

NONE

---

**LineAxis**

```
LineAxis[N] = (XAxis1, YAxis1), (XAxis2, YAxis2), ...;
```

The **LineAxis** parameter defines a set of one or more axes for a line chart, and must correspond to a **DataSet**. The attributes define the specific axis to be used when mapping the X and Y values respectively.
for each data set defined. The XAxis attribute may be set to BOTTOM or TOP, while the YAxis may be set to LEFT or RIGHT.

**Used in These Charts**
Combo, Line, Stock, X-Y

**Example:**
```
LineAxis = (BOTTOM, LEFT), (BOTTOM, LEFT);
```

**Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XAxis</td>
<td></td>
</tr>
<tr>
<td>YAxis</td>
<td></td>
</tr>
</tbody>
</table>

**LineColorTable**

```
LineColorTable[1-50] = Color1, Color2, Color3, Color4, Color5, ...;
```

**Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
</table>
|LineColorTable | defines a set of colors for dataset N that overrides all other color specifications for that set. The parameters used for specifying the color of data points in a chart are (in ascending order of precedence) ColorTable, LinesSets, LineSymbol, and LineColorTable. LineColorTable is used most frequently to color some specific data point.

For example
```
LineColorTable2 = ,,blue;
```

will change the third data point in the second series to blue, while all other datapoints in the chart continue to be colored by one of the other color related parameters.

The colors you can use are defined in the common Color attribute (Chapter 4).

**Used in These Charts**
Combo, Line, Stock, XY

**Example:**
```
BarColorTable2 = ,, red;
```

**Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**LineDropShadow**

```
LineDropShadow = (color, offsetx, offsety, size);
```

**Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
</table>
| LineDropShadow | places a shadow on the background field of the Combo, Line, Stock or XY chart. The color, orientation, and size of the shadow can be defined. The tuple element color sets the color of the shadow. That color value is interpolated to complete transparency as it reaches the end of the shadow’s blur area. Offsetx and offsety define the center point of the shadow; offsetx sets the x-axis offset from
the chart’s center-point; offsety sets the y-axis offset. When an offset attribute is set to a whole number value, the position of the drop shadow is offset from the chart’s center point by the number of pixels set by that whole number. When an offset is set to a fractional value (between 0 and 1), the value is interpreted as a percentage of the width of the object casting the shadow. The center of the drop shadow is repositioned based upon the values or percentages set for offsetx and offsety. Offset attribute values may be positive or negative. Size sets the size of the blur area, the region beyond the actual drop shadow shape where the shadow is extended and blurred into transparency. The size of this blurred region is controlled by the size attribute. The blurred region becomes larger and more diffuse as the value of size is increased. When size is set to a whole number value, the size of the blurred area is defined in pixels. When size is set to a fractional value (between 0 and 1), the value is interpreted as a percentage of the width of the object casting the shadow. When using a fractional value, enclose the value in double-quotes to “escape” the decimal point.

**Used in These Charts**

Combo, Line, Stock, X-Y

**Example:**

```
LineDropShadow = (Black, 5, 5, "0.05");
```

**Attributes**

<table>
<thead>
<tr>
<th>Color</th>
<th>Offsetx</th>
<th>offsety</th>
<th>Size</th>
</tr>
</thead>
</table>

**Color**

Color specifies the base color of the shadow to be drawn behind the chart.

**Example:**

```
LineDropShadow = (black, "-0.05", "-0.05", 55);
```

**Offsetx**

Offsetx specifies the x-coordinate offset from center.

**Example:**

```
LineDropShadow = (black, 25 -10, 25);
```

**Offsety**

Offsety specifies the y-coordinate offset from center.
Example:

$$\text{LineDropShadow} = (\text{black}, 25 \,-10, \, 25);$$

**Size**

Size specifies the width of the blurred area.

Example:

$$\text{LineDropShadow} = (\text{black}, "-.05","-.05", \, 55);$$

**LineFillPattern**

$$\text{LineFillPattern}[N] = (\text{type, color1, color2, imageURL}),...;$$

The LineFillPattern parameter provides a visual pattern fill for the line area of a chart.

<table>
<thead>
<tr>
<th>Type</th>
<th>NONE</th>
<th>no pattern, do default fill, if any</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Built-In Patterns</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FSLASH</td>
<td>front slash type</td>
<td></td>
</tr>
<tr>
<td>BS LASH</td>
<td>back slash type</td>
<td></td>
</tr>
<tr>
<td>DGRID</td>
<td>diagonal grid lines, (front and back slash lines)</td>
<td></td>
</tr>
<tr>
<td>HORIZONTAL</td>
<td>horizontal lines</td>
<td></td>
</tr>
<tr>
<td>VERTICAL</td>
<td>vertical lines</td>
<td></td>
</tr>
<tr>
<td>GRID</td>
<td>grid lines, (horizontal and vertical lines)</td>
<td></td>
</tr>
<tr>
<td><strong>Gradients</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRADIENTVERTICAL</td>
<td>bottom to top style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTHORIZONTAL</td>
<td>left to right style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTFDIAG</td>
<td>top right to bottom left style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTBDIAG</td>
<td>top left to bottom right style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTRADIAL</td>
<td>radial style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTCENTERHORIZONTAL</td>
<td>center out horizontal style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTCENTERVERTICAL</td>
<td>center out vertical style gradient</td>
<td></td>
</tr>
<tr>
<td><strong>Images</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IMAGE</td>
<td>use an image specified in the optional imageURL element</td>
<td></td>
</tr>
</tbody>
</table>

**color1**

This color is used in the following ways:
- Foreground color for patterns
- Starting color for gradients
- Ignored in images

**color 2**

This color is used in the following ways:
Common CDL Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>imageURL</code></td>
<td>The URL to an image to use as the fill</td>
</tr>
</tbody>
</table>

### Used in These Charts

**Line**

**Example:**

```java
LineFillPattern = (VERTICAL, blue, white);
```

**Attributes**

<table>
<thead>
<tr>
<th>Type</th>
<th>Color1</th>
<th>Color2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageFormat</td>
<td>ImageURL</td>
<td></td>
</tr>
</tbody>
</table>

**LineLabels[n]**

`LineLabels[1-50] = ("Label", "URL", "Target"),...;`

*LineLabels* defines sets of sets of active label destinations for the lines in a line graph, and maps to LineSets, which must also be specified.

**Used in These Charts**

Combo, Stock

**Example:**

```java
LineLabels1 = ("OUTLINE", "fr71h.html", "frame1"), ("OUTLINE", "fr72h.html", "frame1"), ("OUTLINE", "fr73h.html", "frame1");
```

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
</table>

**LineSet[n]**

`LineSet[1-50] = y1, y2, y3,...;`

Defines a list of from 1 up to 50 Y values for each line set in a line chart, and corresponds to the LineSets parameter, with which it must be used. X values are not specified because the line chart style assumes automatic uniform spacing along the X Axis of the graph.
Common CDL Attributes

Used in These Charts
Combo, Line, Radar, Stock, X-Y

Example:
```csharp
LineSets = ("Hammers", black), ("Clamps", black), ("Wrenches", black),
           ("Pliers", black);
LineSet1 = 40, -100, 80, 50, 40;
LineSet2 = 60, 50, 10, 40, 30;
LineSet3 = -50, 20, 49, 10, 89;
LineSet4 = 40, 50, 150, 150, 200;
```

Attributes

Y

Y

A real value, corresponding to a set within the LineSet parameter. Null values, which consist of no space between delimiting commas, may be assigned and are displayed differently depending on the GraphType.

Example:
```csharp
LineSet1 = 10, 20, 30, 40;
LineSet2 = 60,,10,40,30;
```

Values

>= 0

Default

None

LineSets

LineSets[N] = (Name1, SymColor1), (Name2, SymColor2), ...

By default, the Name and SymColor assigned to each data set will be used in the chart legend. At most 50 line sets may be displayed.

Used in These Charts
Combo, Line, Radar, Stock, X-Y

Example:
```csharp
LineSets = ("Sprocket", black),
           ("Actuator", black),
           ("Do-Hicky", black),
           ("Thingy", black);
```

Attributes

Name  SymColor
Name

The Name of a LineSet is an identifier, which will only appear in the chart if a legend is used in the chart.

Example:

```plaintext
LineSets = ("Sprocket", black),
           ("Actuator", black),
           ("Do-Hicky", black),
           ("Thingy", black);
```

Value

Any legal string value

Default

None

SymColor

If a SymColor is not specified in the vector, then the previously specified color will be used. If the color is specified as NULL, then a default color will be chosen from the color table. Symcolor is used in the definitions of a legend, and may be overridden by the specification of a color in the LineStyle parameter.

Example:

```plaintext
LineSets = ("Sprocket", black),
           ("Actuator", purple),
           ("Do-Hicky", red),
           ("Thingy", orange);
```

Value

Any legal color value (see Color attribute in Chapter 4).

Default

black

LineStyle

LineStyle = (Type, LineWidth, Color, FillColor, LineType, FillType), ... ;

This parameter specifies the line style to be displayed for one or more line sets.

Used in These Charts

Bubble, Combo, Line, Radar, Stock, Strip, X-Y

Example:

```plaintext
LineStyle = (SOLID, 3, blue, blue, NORMAL, OFF),
           (SOLID, 2, red, red, FIT, COLOR);
```
**Attributes**

<table>
<thead>
<tr>
<th>Color</th>
<th>FillColor</th>
<th>LineType</th>
<th>LineWidth</th>
<th>Type</th>
</tr>
</thead>
</table>

**FillColor**

If this attribute is not NULL, then the area under the line set will be filled with the given color.

**Example:**

```plaintext
LineStyle = (SOLID, 3, skyblue, lightskyblue),
           (SOLID, 3, red, pink), <!-- red line with pink fill -->
           (SOLID, 3, green, lightgreen),
           (SOLID, 3, orange, peachpuff);
```

**Values**

- **NULL** Also, value left unspecified: No color fills the area under the line.
- Any legal color Area under the line is filled. See Chapter 4 for the Color attribute.

**Default**

None

**Type**

The style of line to draw

**Values**

- **NONE**
- **SOLID**
- **DOTTED**
- **DASHED**
- **DOTDASH**

**Default**

SOLID

**LineType**

The type of line to use to connect the points in the series

**Values**

- **NORMAL**
- **FIT**
- **BOTH**

**Default**

NORMAL

**FillType**
The type of fill to use for the series. ON will force a fill, using a color from the Color Table if a fill color is not defined. OFF will not fill, even if a fill color is specified, COLOR will will only if a fill color is specified.

**Values**

ON  
OFF  
COLOR

**Default**

COLOR

---

**LineSymbol**

```
LineSymbol[N] = (Type, Size, Style, BorderColor, BorderWidth, ImageURL, SymbolColor,ShadowWidth),...;
```

LineSymbol specifies the symbols to be displayed for one or more line sets. That is, the first parenthesized group defines the symbol for the first line set, and so on.

**Used in These Charts**

Combo, Line, Radar, Stock, Strip, X-Y

**Example:**

```
LineSymbol = (IMAGE,,,,,"$SYMBOLS/save.gif"),
(IMAGE,,,,,"$SYMBOLS/cut.gif"),
(IMAGE,,,,,"$SYMBOLS/paste.gif"),
(IMAGE,,,,,"$SYMBOLS/pinwheel.gif");
```

```
LineSymbol = (CIRCLE, 6, BOTH, white, l,grey,0),
(SQUARE, 6, BOTH, cyan, 1,red,2),
(DIAMOND, 6, BOTH, firebrick, 1),
(CROSS, 6, BOTH, green, 1);
```

**Attributes**

<table>
<thead>
<tr>
<th>BORDERCOLOR</th>
<th>BORDERWIDTH</th>
<th>IMAGEURL</th>
<th>SYMBOLCOLOR</th>
<th>SHADOWWIDTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIZE</td>
<td>STYLE</td>
<td>SYMBOLCOLOR</td>
<td>TYPE</td>
<td></td>
</tr>
</tbody>
</table>

---

**Size**

Size specifies the size of the symbol in pixels. This attribute is ignored for IMAGE symbols.

**Example:**

```
LineSymbol = (CIRCLE, 6, BOTH, white, l),
(SQUARE, 7, BOTH, cyan, 1),
(DIAMOND, 9, BOTH, firebrick, 1),
(CROSS, 8, BOTH, green, l);
```
**Values**
Any integer value in pixels

**Default**
None

**SymbolColor**

SymbolColor specifies for foreground color of the LineSymbol.

**ShadowThickness**

ShadowThickness specifies for size of the shadow behind a LineSymbol. Any number other than 0 causes NetCharts to choose a shadow size based on the size of the symbol. Specifying a value of 0 suppresses the shadow.

**Style**

Style specifies how the LineSymbol should be drawn, including FILLED, OUTLINED, or BOTH. If FILLED is specified, the symbol is filled with the line set color. If OUTLINED is specified, only the outline is drawn, using the line set color. If BOTH is specified, then the symbol is filled with the line set color and the outline is drawn using the borderColor.

**Example:**
```
LineSymbol = (CIRCLE, 6, BOTH, white, red),
              (SQUARE, 6, OUTLINE, orchid),
              (DIAMOND, 6, FILLED),
              (CROSS, 6, BOTH, white, darkcyan);
```

**Values**

FILLED Symbol is filled with the LineSet color.
OUTLINED Only the outline is drawn, using the LineSet color
BOTH Symbol is filled with the LineSet and the outline is drawn using theBorderColor

**Default**
None

**Type**

Type specifies the type of symbol to be displayed on the line set.

**Example:**
```
LineSymbol = (CIRCLE, 6, BOTH, white, 1),
              (SQUARE, 6, BOTH, cyan, 1),
              (DIAMOND, 6, BOTH, firebrick, 1),
              (CROSS, 6, BOTH, green, 1);
```
Values

<table>
<thead>
<tr>
<th>Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>No symbol is displayed.</td>
</tr>
<tr>
<td>CIRCLE</td>
<td>Displays circles</td>
</tr>
<tr>
<td>SQUARE</td>
<td>Displays squares</td>
</tr>
<tr>
<td>DIAMOND</td>
<td>Displays diamonds</td>
</tr>
<tr>
<td>CROSS</td>
<td>Displays crosses</td>
</tr>
<tr>
<td>TARGET</td>
<td>Displays targets (bulls-eye)</td>
</tr>
<tr>
<td>TRIANGLEDOWN</td>
<td>Displays downward pointing triangles</td>
</tr>
<tr>
<td>TRIANGLEUP</td>
<td>Displays upward pointing triangles</td>
</tr>
<tr>
<td>IMAGE</td>
<td>If specified, the ImageURL attribute is required and will be used to load a GIF image for the symbol.</td>
</tr>
</tbody>
</table>

Default

None

LineSymbolSpotlights

LineSymbolSpotlights = (start.stop,center,centeroffsetx,centeroffsety, focusoffsetx, focusoffsety, radius)...;

Adds or overlays a color fill over one or more existing fill patterns to produce multiple layered color effects. The spotlight “illuminates” the line symbols. The center of the spotlight and its focus can be adjusted independently by adjusting offsets from the symbol center point. The elements centeroffsetx and centeroffsety set the x and y-coordinates of the center of the spotlight as an offset of the chart center point. When set to whole numbers, centeroffsetx and centeroffsety specify the number of pixels to offset from the chart center point. When set to fractional values (between 0 and 1), they are interpreted as percentages of the width of the symbols. Percentage values are written using a decimal point (e.g. “0.05” and “0.50” represent 5% and 50%, respectively). When using a fractional value, enclose the tuple element in double-quotes to “escape” the decimal point. The element focusoffsetx is the offset from the chart center which defines the x-coordinate of the focus point of the spotlight. The element focusoffsety is the offset from the chart center which defines the y-coordinate of the focus point of the spotlight. When set to whole numbers, focusoffsetx and focusoffsety specify the offset from the center in pixels. When set to fractional values (between 0 and 1), they are interpreted as percentages of the width of the bars. The element radius sets the size of the spotlight, from its center to its edge. When set to a whole number, it sets the size of the radius in pixels; when set to a fractional value, it sets the radial diameter of the spotlight based upon that percentage of the minimum height and width of the symbols on the chart.

Used in These Charts

Bar, Combo, Pareto, Stock, XY

Example:

```
LineSymbolSpotlights = (purple_40,blue_155,RIGHT,50, -50,100,150,250);
```

Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>start</th>
<th>stop</th>
<th>center</th>
</tr>
</thead>
<tbody>
<tr>
<td>centeroffsetx</td>
<td>centeroffsety</td>
<td>focusoffsetx</td>
<td>radius</td>
</tr>
</tbody>
</table>

CDL Reference Manual – 7.1

© Visual Mining, Inc 2002-2011

www.visualmining.com
**Start**

*Start* specifies the first of the two colors which will be interpolated to produce a gradient spotlight.

**Example:**

\[\text{LineSymbolSpotlights} = (\text{purple}_40, \text{blue}_155, \text{LEFT}, 20,10,120,-120,250);\]

**Stop**

*Stop* specifies the second of two colors which will be interpolated to produce a gradient spotlight.

**Center**

*Center* specifies the position around of the center of the chart where the spotlight center will be placed.

**Values**

- **RIGHT**: Offsets the center point of the spotlight to the right of the center point of the chart.
- **LEFT**: Offsets the center point to the left.
- **TOP**: Offsets the center point to the top.
- **BOTTOM**: Offsets the center point to the bottom.
- **CENTER**: Uses the chart center point for the spotlight center point.
- **TOPRIGHT**: Offsets the center point of the spotlight to the top right.
- **TOpleft**: Offsets the center point of the spotlight to the top left.
- **BOTomright**: Offsets the center point of the spotlight to the bottom right.
- **BOTomleft**: Offsets the center point of the spotlight to the bottom left.

**Default**

CENTER

**Centeroffsetx**

*Centeroffsetx* specifies the x-coordinate offset for the spotlight center.

**Centeroffsety**

*Centeroffsety* specifies the y-coordinate offset for the spotlight center.
### Focusoffsetx

Focusoffsetx specifies the x-coordinate offset for the center of the spotlight’s focus.

### Focusoffsety

Focusoffsety specifies the y-coordinate offset for the center of the spotlight’s focus.

### Radius

Radius specifies the length of the radius of the spotlight from the center of the spotlight.

### LineValueLabel

LineValueLabel[N] = (mode, color, font name, width);

Defines the label value to be displayed for each point in a lineset.

**Used in These Charts**
Bar, Combo, Line, Radar, Stock, Strip, X-Y

**Example:**
```plaintext
LineValueLabel = ("ON", black, "Helvetica", 18);
```

**Attributes**
Mode  Color  Font Name  Width

### LineValueLabelBox

LineValueLabelBox[N] = (color, mode, depth);

Defines the line label box to be displayed with each point on a lineset.

**Used in These Charts**
Bar, Combo, Line, Radar, Stock, Strip, X-Y

**Example:**
```plaintext
LineValueLabelBox = (grey, RAISED, 3);
```
Common CDL Attributes

**Attributes**

Color    Mode    Depth

**LineValueLabelStyle**

LineValueLabelStyle[N] = labelposition1, labelposition2, ... labelpositionN;

Defines where the LineValueLabel text will display for each point of the lineset.

**Used in These Charts**

Bar, Combo, Line, Radar, Stock, Strip, X-Y

**Example:**

```
LineValueLabelStyle = TOP;
```

**Values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOPLEFT</td>
<td>Displayed at the top left point in a lineset</td>
</tr>
<tr>
<td>TOP</td>
<td>Displayed at the top of the point in a lineset</td>
</tr>
<tr>
<td>TOPRIGHT</td>
<td>Displayed at the top right point in a lineset</td>
</tr>
<tr>
<td>LEFT</td>
<td>Displayed at the left of the point in a lineset</td>
</tr>
<tr>
<td>CENTER</td>
<td>Displayed at the center point in a lineset</td>
</tr>
<tr>
<td>RIGHT</td>
<td>Displayed at the right of the point in a lineset</td>
</tr>
<tr>
<td>BOTTOMLEFT</td>
<td>Displayed at the bottom left point in a lineset</td>
</tr>
<tr>
<td>BOTTOM</td>
<td>Displayed at the bottom of the point in a lineset</td>
</tr>
<tr>
<td>BOTTOMRIGHT</td>
<td>Displayed at the bottom right point in a lineset</td>
</tr>
</tbody>
</table>

**Default**

TOP

**Attributes**

Label Position

**LineWidth**

LineWidth[N] = PercentDepth;

LineWidth determines the width of a 3D line in a line chart, with reference to the amount of space specified by the z-dimension in the Line3DDepth parameter.

**Used in These Charts**

Combo, Line, Pareto, Stock, X-Y
Example:

```
LineWidth = 60;
```

Attributes

PercentDepth

**PercentDepth**

This percentage value, 0-100 or 0.0-1.0, determines the amount of space actually used to display 3D lines when Line3DDepth is greater than 1.

Example:

```
LineWidth = 0.6;
```

Values

0 to 100 or 0.0 to 1.0  If 100 percent is specified, then each line will completely fill the amount of space specified by Line3DDepth in the Z dimension. If 50 percent is specified, then each line will occupy only 50 percent of that space, which will result in a visible separation between each line set.

Default

100

**MeanActiveLabels**

```
MeanActiveLabels = ("Labell", "URL1", "Target1"),...;
```

MeanActiveLabels define the active labels associated with mean values.

**Used in These Charts**

Box Chart

Example:

```
MeanActiveLabels = (lightgray, SHADOW, 3,,gray);
```

Attributes

Label, Target, URL

**MeanColor**

```
MeanColor = Color;
```
MeanColor allows users to specify the color to be used to display the mean value. This value is used if no color is specified in the MeanSymbol parameter. The default value is the value of the MedianColor.

**Used in These Charts**
- **Box Chart**

**Example:**
```
MeanColor = red;
```

**Attributes**

Color

---

**MeanLine**

MeanLine = (type, width, color);

MeanLine allows the users to draw a line that connects the mean of each data series in a Box Chart.

**Used in These Charts**
- **Box Chart**

**Example:**
```
MeanLine = (DASHED, 3, green);
```

<table>
<thead>
<tr>
<th>type</th>
<th>type of line to draw. Legal values are SOLID, DASHED, DOTTED and DOTDASH</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>width in pixels of the line, the default is 1.</td>
</tr>
<tr>
<td>color</td>
<td>color of the line, the default is the MeanColor</td>
</tr>
</tbody>
</table>

**Attributes**

Color, Type, Width

---

**MeanSymbol**

MeanSymbol = (type1, size1, style1, bordercolor1, borderwidth1, image1, color1),...;
MeanSymbol is used to define the style in which to draw the mean value.

**Used in These Charts**

**Box Chart**

**Example:**

MeanSymbol = (CIRCLE, 6, BOTH, white, 1);

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeN</td>
<td>the type of symbol to use for the mean in data series N. Legal values are NONE, CIRCLE, SQUARE, DIAMOND, CROSS, TARGET, TRIANGLEDOWN, TRIANGLEUP, IMAGE</td>
</tr>
<tr>
<td>sizeN</td>
<td>the size in pixels of the symbols for the mean in data series N</td>
</tr>
<tr>
<td>styleN</td>
<td>the drawing style for the mean in data series N. Legal values are FILLED, OUTLINED or BOTH</td>
</tr>
<tr>
<td>borderColorN</td>
<td>the color of the border for the mean in data series N</td>
</tr>
<tr>
<td>borderWidthN</td>
<td>the width in pixels of the border for the mean in data series N</td>
</tr>
<tr>
<td>imageN</td>
<td>the image to use for displaying for the mean in data series N</td>
</tr>
<tr>
<td>colorN</td>
<td>the color for the mean in data series N</td>
</tr>
</tbody>
</table>

**Attributes**

Type, Size, Style, borderColor, borderWidth, Image, Color

**MedianColor**

MedianColor[N] = Color;

MedianColor determines the color to be used when drawing the median. If MedianColor is not defined, the default color is white.

**Used in These Charts**

**Box**

**Example:**

MedianColor = xE3E3E3;
MedianColor = silver;

**Attributes**

Color
MetaData

MetaData = ("name", "value"), ("name", "value"), ...

The MetaData parameter allows a chart writer to embed useful information about the chart within its definition.

Used in These Charts

ALL

Example:

The parameter allows the setting of any number of name/value pairs. For example,

        MetaData=("Author", "John Doe"),
                ("Creation Date", "January 21, 2001"),
                ("Department", "Sales");

This information, once embedded in the chart through the MetaData parameter, can be searched via a NetCharts Server. For example, let's say you want to search your server for all charts created for the sales department. A possible solution would be to institute a company wide policy mandating that all charts have a MetaData parameter containing the department for which the chart was developed. Searching for the word Department and Sales in all charts would give the desired results.

Attributes

Name     Value

MinimumDataPoints

MinimumDataPoints = int_val;

MinimumDataPoints defines the number of data points the must be present in a series in order for the chart to draw in summary mode. IF ShowDataPoints is OFF, and the minimum number of data points necessary to calculate a summary display are not present, the raw data will be displayed.

Used in These Charts

Box Chart

Example:

        MinimumDataPoints = 12;
Attributes
Value

NaturalDisplayOrder

NaturalDisplayOrder = $ON | OFF$;

The new Box Chart provides control over the left to right order of data set displays. The default behavior for a BoxChart in VERTICAL mode is to display the series from right to left. Most charts make more sense when they are laid out left to right. The NaturalDisplayOrder parameter provides control over this behavior.

Used in These Charts
Box Chart

Example:

```
NaturalDisplayOrder = ON;
NaturalDisplayOrder = OFF;
```

Default
ON

Attributes
Node

NodeBox

NodeBox[N] = ($Color$, $BorderType$, $BorderWidth$, $ImageURL$, $ImageFormat$, $BorderColor$, $TRCornerStyle$, $BRCornerStyle$, $BLCornerStyle$, $CornerColor$), ...;

The NodeBox parameter controls the appearance of the box that forms the nodes in diagram charts. If only one ‘tuple is used, all the boxes will appear the same. Otherwise, the order of the ‘tuples maps to the order in which the nodes were defined in the Nodes parameter.

Notice that in the example, a one-pixel transparent GIF image is used to invisibly anchor two edges so that the edges may look as though they turn a corner. This is done because the Edge parameter only allows one single straight line per edge.

Used in These Charts
Diagram
Example:

NodeBox =
    (white, BOX, 1, "../../../images/blonde-woman.gif",gray),
    (white, BOX, 1, "../../../images/susan.gif",gray),
    (white, BOX, 1, "../../../images/glasses-girl.gif",gray),
    (white, BOX, 1, "../../../images/tie-guy.gif",gray),
    (white, BOX, 1, "../../../images/coffeeshopguy.gif",gray),
    (white, NONE, 0, "../../../images/pixel-clear.gif"),
    (white, NONE, 0, "../../../images/pixel-clear.gif"),
    (white, NONE, 0, "../../../images/pixel-clear.gif");

Attributes

<table>
<thead>
<tr>
<th>Color</th>
<th>BorderType</th>
<th>BorderWidth</th>
<th>ImageURL</th>
<th>ImageFormat</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BorderColor</td>
<td>TRCornerStyle</td>
<td>BRCornerStyle</td>
<td>BLCornerStyle</td>
<td>CornerColor</td>
</tr>
</tbody>
</table>

XXCornerStyle

The drawing style for each of the four corners of the region. Styles are specified in a clockwise fashion starting in the upper left - Top Left, Top Right, Bottom Right, and Bottom Left. Legal values are SQUARE, SNIP and ROUND. The default is SQUARE.

NodeDrag

NodeDrag[N] = "ON"|"OFF";

The NodeDrag switch is used to allow or stop the user from dragging the nodes of the diagram chart with the mouse.

Used in These Charts

Diagram

Example:

NodeDrag = "ON";
NodeDrag = "OFF";

Attributes

(Switch)

Switch

This switch sets the on/off state.

Example:

NodeDrag = "ON";
NodeDrag = "OFF";

CDL Reference Manual – 7.1

© Visual Mining, Inc 2002-2011
www.visualmining.com
Values

ON
Allows the user to drag the hands on the applet dial

OFF
Stops the user from dragging the hands on the applet dial

Default
ON

NodeLabel

$\text{NodeLabel}[N] = (\text{Color}, \text{FontName}, \text{FontSize}, \text{Angle}), \ldots$

NodeLabel is an optional parameter that controls the specific appearance and style of the labels for named nodes defined with the Node parameter. The order of the node labels maps to the order in which the nodes were defined. If any attribute is not specified in the vector, then the previously specified attribute will be used.

Note that if any attribute is not specified in the ‘tuple, then the previously specified attribute will be used. If a color is specified as NULL or omitted, then a color will be chosen from a ColorTable parameter.

Used in These Charts

Diagram

Example:

NodeLabel = (black, "Helvetica", 8);

NodeLabel = (yellow,"TimesRoman",8,0),(yellow,"TimesRoman",8,0);

Attributes

Label | Color | FontName | FontSize | Angle | interiorAlignment

Nodes

$\text{Nodes}[N] = (\text{Name}, \text{Label}, X, Y), \ldots$

Nodes defines all the nodes in a diagram chart, by naming, labeling, and positioning them. The example below was used in the diagram chart on the right.

Used in These Charts

Diagram

Example:

Nodes =
("CEO", ",", 200, 31),
("VP Sales","", 31, 150),
("COO", ",", 200, 150),
The $x$ value maps the location of the center of a node along the chart’s X-axis, and is given in pixels.

**Example:**

```plaintext
Nodes =
("CEO", "", 200, 31),
("VP Sales", "", 31, 150),
("COO", "", 200, 150),
("VP Marketing", "", 350, 150),
("Webmaster", "", 350, 240)
("sales-node", "", 31, 100), <!-- Invisible node to turn a corner -->
("coo-node", "", 200, 100), <!-- Invisible node to turn a corner -->
("marketing-node", "", 350, 100); <!-- Invisible node to turn a corner -->
```

**Values**

Positive numbers greater than or equal to 0.

**Default**

No defaults

The $y$ value maps the location of the center of a node along the chart’s Y-axis, and is given in pixels.

**Example:**

```plaintext
Nodes =
("CEO", "", 200, 31),
("VP Sales", "", 31, 150),
("COO", "", 200, 150),
("VP Marketing", "", 350, 150),
("Webmaster", "", 350, 240)
("sales-node", "", 31, 100), <!-- Invisible node to turn a corner -->
("coo-node", "", 200, 100), <!-- Invisible node to turn a corner -->
("marketing-node", "", 350, 100); <!-- Invisible node to turn a corner -->
```

**Values**

Positive numbers greater than or equal to 0.

**Default**

No defaults
NoteActiveLabels[n]

NoteActiveLabels[1-20] = ("Label", "URL", "Target"), ...;

NoteActiveLabels[n] defines a list of custom active labels to be associated with each note in a particular NoteSet. That is, these labels will be displayed whenever the mouse "dwells" over a given note. This can be used to provide the user with additional information about that particular note or to "drill-down" to another document. See ActiveLabels for a detailed explanation of active label capabilities.

Used in These Charts

All

Example:

NoteActiveLabels1 = ("$13,422", ".../daily/stats.html", "statwin"), ("$27,002"), 
("$33,812"), ("$12,799");

Attributes
Label URL Target

NoteArrow

NoteArrow[N] = (LineType1, LineWidth1, LineColor1, ArrowType1, ArrowStyle1), (LineType2, LineWidth2, 
LineColor2, ArrowType2, ArrowStyle2),...;

NoteArrow defines a list of list of line and arrow definitions, with the first in the list referring to NoteSet1, the second in the list to NoteSet2, and so on.

Used in These Charts

All

Example:

NoteArrow = (SOLID, 2, red, TOFROM, BLOCK);

Attributes
LineType LineWidth LineColor ArrowType ArrowStyle

LineColor

LineColor controls the color of the line in the NoteArrow. This parameter operates in all respects as the typical Color parameter. See Chapter 4 for specifics.

ArrowType

ArrowType controls the direction of the arrowhead terminating the NoteArrow.
Example:

```
NoteArrow = (SOLID, 2, red, TOFROM, BLOCK);
```

Values

- **NONE**: No arrowhead is shown
- **FROMTO**: Arrowhead is oriented and placed with last point on the line
- **TOFROM**: Arrowhead is oriented and placed with first point on the line
- **BOTH**: Two arrowheads are used, one oriented and placed with the last point on the line, the second oriented and placed with the first point on the line.

Default

NONE

**ArrowStyle**

ArrowStyle controls the shape of the arrowhead terminating the NoteArrow.

Example:

```
NoteArrow = (SOLID, 2, red, TOFROM, BLOCK);
```

Values

- **ROUND**: Round arrowhead
- **SHARP**: Conventional triangular pointed arrowhead
- **BLOCK**: Square arrowhead

Default

SHARP

**NoteAxis**

```
NoteAxis[N] = (XAxis1, YAxis1), (XAxis2, YAxis2), ...;
```

NoteAxis defines a list of axis mapping pairs, with the first in the list referring to NoteSet1, the second in the list to NoteSet2, and so on.

When an X and Y value must be mapped to screen pixels, the axis mapping pair for that NoteSet is used to determine the mapping strategy.

The XAxis and YAxis parameters may be of different types.

**Used in These Charts**

All

Example:

```
NoteAxis = (PERCENT, LEFT), (BOTTOM, LEFT), (BOTTOM, LEFT);
```

© Visual Mining, Inc 2002-2011
www.visualmining.com
**Attributes**

**XAxis**

XAxis controls to which location the Notes’ X values are relative.

**Example:**

\[ \text{NoteAxis} = (\text{PERCENT, PERCENT}), (\text{BOTTOM, LEFT}), (\text{PIXEL, LEFT}); \]

**Values**

- **BOTTOM**: X value is relative to bottom axis values
- **TOP**: X value is relative to top axis values
- **PERCENT**: X value is a percentage of window width (0-100%)
- **PIXEL**: X value is an absolute pixel location relative to the left edge of the applet

**Default**

BOTTOM

**YAxis**

YAxis controls to which location the Notes’ Y values are relative.

**Example:**

\[ \text{NoteAxis} = (\text{PERCENT, PERCENT}), (\text{BOTTOM, LEFT}), (\text{PIXEL, PIXEL}); \]

**Values**

- **LEFT**: Y value is relative to left axis values
- **RIGHT**: Y value is relative to right axis values
- **PERCENT**: Y value is a percentage of window height (0-100%)
- **PIXEL**: Y value is an absolute pixel location relative to the top edge of the applet

**Default**

LEFT

**NoteBox**

\[ \text{NoteBox}[N] = (\text{Color1, BorderType1, BorderWidth1, ImageURL1, ImageFormat1, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor}), \ldots; \]

The NoteBox parameter is a list of region definitions, with the first in the list referring to NoteSet1, the second in the list to NoteSet2, and so on. The NoteBox, if defined, is drawn underneath the note label and can be used to highlight the note.
**Common CDL Attributes**

**XXCornerStyle**

The drawing style for each of the four corners of the region. Styles are specified in a clockwise fashion starting in the upper left - Top Left, Top Right, Bottom Right, and Bottom Left. Legal values are SQUARE, SNIP and ROUND. The default is SQUARE.

**NoteDrag**

NoteDrag[N] = ON | OFF;

The NoteDrag feature allows NoteSets to be individually configured to allow, or dis-allow dragging.

**Example:**

```plaintext
NoteDrag = "ON";
```

**Attributes**

Values

- **ON** Allows the user to drag NoteSets
- **OFF** Stops the user from dragging NoteSets

**Default**

- **ON**

**NoteLabel**

NoteLabel[N] = ("Label1", "Color1", "FontName1", FontSize1, Angle1), ...;

The NoteLabel parameter contains a list of label definitions, with the first parenthesized group in the list referring to NoteSet1, the second parenthesized group defining NoteSet2, and so on.
If \texttt{Mode} is \texttt{OFF}, then the note text for that \texttt{NoteSet} will not be shown. In any other case, the \texttt{NoteSet} text will be drawn using the given label attributes in the \texttt{NoteLabel} parameter.

<table>
<thead>
<tr>
<th>interiorAlignment</th>
<th>Specifies the alignment to use in text strings that contain multiple lines.</th>
</tr>
</thead>
</table>

The legal values for \texttt{interiorAlignment} are \texttt{LEFT}, \texttt{RIGHT}, or \texttt{CENTER}.

**Used in These Charts**

All

**Example:**

\begin{verbatim}
NoteLabel = ("ON", black, Helvetica, 10), ("ON", black, Helvetica, 10),
            ("ON", black, Helvetica, 10);

NoteLabel = ("", black, "Dialog", 12), ("", x5E8109, "Helvetica", 12), ("",
            x1A4C8F, "Helvetica", 12, 270);
\end{verbatim}

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>Color</th>
<th>FontName</th>
<th>FontSize</th>
<th>Angle</th>
<th>interiorAlignment</th>
</tr>
</thead>
</table>

**NoteSet[n]**

\begin{verbatim}
NoteSet1 = ("Text1", X, Y, X1, Y1, X2, Y2, X3, Y3), ...;
\end{verbatim}

For each defined \texttt{NoteSet} defined in the \texttt{NoteSets} parameter, there should be a corresponding \texttt{NoteSet[n]} where \texttt{[n]} refers to the appropriate \texttt{NoteSet} number.

Each \texttt{NoteSet} may have an unlimited number of notes, each with its own text string, X-Y position vector, and optional point vectors to draw an arrow segment. While the X-Y vector must be defined, the X1-Y1, X2-Y2, and X3-Y3 vectors are optional and are used to draw a line segment, if desired.

**Used in These Charts**

All

**Example:**

\begin{verbatim}
NoteSet1 = ("x", 95, 35, 5, 35), ("y", 50, 90, 50, 25);

NoteSet2 = ("L", 48, 65), ("a", 65, 33), ("0", 47, 29), ("f(x) = L", 65, 25),
            ("\lim_{nx \to a}", 38, 25);
\end{verbatim}

**Attributes**

<table>
<thead>
<tr>
<th>Text</th>
<th>X</th>
<th>Y</th>
<th>X1</th>
<th>Y1</th>
<th>X2</th>
<th>Y2</th>
<th>X3</th>
<th>Y3</th>
</tr>
</thead>
</table>

**Text**

\texttt{Text} displays the text of the \texttt{NoteSet}. The text may contain multiple lines, delimited by the \texttt{\n} (new line) symbol.
Example:

```
NoteSet1 = ("The Amplifier clearly led\nall components in overall\nsales. Bob McIness and \nstaff did a great job.",82,35,68,70);
```

Values

Any alphanumeric text. The text may contain multiple lines, delimited by the \n(new line) symbol. Text may also be a null, or empty string, "".

Default

None

X, Y

X and Y are required. These first two attributes control the note’s relative position. This location vector is also used as the first point for the arrow line if X1-Y1 are defined.

Example:

```
NoteSet1 = ("The Amplifier clearly led\nall components in overall\nsales. Bob McIness and \nstaff did a great job.",82,35,68,70);
```

Values

Any real numbers. What you use also depends on what you defined with the NoteAxis parameter.

Default

None

X1, Y1

X1 and Y1 are optional, and define either a corner or an endpoint for an arrow line for their note.

Example:

```
NoteSet1 = ("The Amplifier clearly led\nall components in overall\nsales. Bob McIness and \nstaff did a great job.",82,35,68,70);
```

Values

Any real numbers. What you use also depends on what you defined with the NoteAxis parameter.

Default

None

X2, Y2

X2 and Y2 are optional, and define either a corner or an endpoint for an arrow line for their note.
Example:

NoteSet4 = ("Sudden Gains", .5, 45, 0, 55, 10, 55, 10, 65);

Values
Any real numbers. What you use also depends on what you defined with the NoteAxis parameter.

Default
None

X3, Y3

X3 and Y3 are optional, and define the endpoint for an arrow line for their note.

Example:

NoteSet4 = ("Sudden Gains", .5, 45, 0, 55, 10, 55, 10, 65);

Values
Any real numbers. What you use also depends on what you defined with the NoteAxis parameter.

Default
None

NoteSets

NoteSets[N] = ("Name1", Justify1), ("Name2", Justify2), ...;

The NoteSets parameter is a required parameter for displaying notes, and contains a collection of NoteSet names and justification types. The number of defined NoteSets[n] is based on the number of parenthesized groups, or tuples, defined herein.

Used in These Charts
All

Example:

NoteSets = ("note1");

NoteSets = ("Axes", BOTTOM), ("TextOnly", CENTER), ("Text2Only", TOP);

Attributes

Name Justify

Name is the identifying name for the NoteSet.
Example:

```
NoteSets = ("Axes", BOTTOM), ("TextOnly", CENTER), ("Text2Only", TOP);
```

Values

Simple alphanumeric text, which will not be seen by the user.

Default

None. Must have a name.

Justify

Justify defines where each note is placed relative to the note’s X-Y location, which is defined in the NoteSet[n] parameter.

Example:

```
NoteSets = ("Axes", BOTTOM), ("TextOnly", CENTER), ("Text2Only", TOP);
```

Values

- **TOPLEFT**: The note’s top left corner becomes its origin
- **TOP**: A point horizontally centered on the note’s top edge becomes its origin
- **TOPRIGHT**: The note’s top right corner becomes its origin
- **LEFT**: A point vertically centered on the left side becomes its origin
- **CENTER**: The center of the note becomes its origin
- **RIGHT**: A point vertically centered on the right side becomes its origin
- **BOTTOMLEFT**: The note’s bottom left corner becomes its origin
- **BOTTOM**: A point horizontally centered on the note’s bottom edge becomes its origin
- **BOTTOMRIGHT**: The note’s bottom right corner becomes its origin

Default

CENTER

NumberFormat

```
NumberFormat = ("decimalSymbol", "groupSymbol", groupSize);
```

The NumberFormat parameter can be used to specify the symbols and group size used when formatting numeric values. The default decimalSymbol is ".", the default groupSymbol is "," and the default groupSize is 3.

Used in These Charts

All
**Example:**

```
NumberFormat = ("," , ".");
```

would generate the following results for the given format for the decimal value 1234.456:

<table>
<thead>
<tr>
<th>Format</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>%f</code></td>
<td>1234.46</td>
</tr>
<tr>
<td><code>%f</code></td>
<td>1.234,46</td>
</tr>
<tr>
<td><code>%d</code></td>
<td>1234</td>
</tr>
<tr>
<td><code>%d</code></td>
<td>1.234</td>
</tr>
<tr>
<td><code>%8,1f</code></td>
<td>1.234,5</td>
</tr>
<tr>
<td><code>%08,1f</code></td>
<td>001234,5</td>
</tr>
</tbody>
</table>

**Attributes**

decimalSymbol, groupSize, groupSymbol

**OutlierActiveLabels**

```
OutlierActiveLabels = ("Label1", "URL1", "Target1")....;
```

OutlierActiveLabels define the active labels associated with mean values.

**Used in These Charts**

- Box Chart

**Example:**

```
OutlierActiveLabels = ("Exception condition","javascript:doException()");
```

**Default**

**Attributes**

Label, URL, Target

**OutlierColor**

```
OutlierColor[N] = Color;
```

OutlierColor controls the color to be used when drawing outliers in a box chart. This value is used if no color is specified in the OutlierSymbol parameter. The default value is the value of MedianColor.
Common CDL Attributes

Used in These Charts
Box

Example:
OutlierColor = xE3E3E3;
OutlierColor = silver;

Attributes
Color

OutlierSymbol

OutlierSymbol = (type1, size1, style1, bordercolor1, borderwidth1, image1, color1),...;

OutlierSymbol is used to define the style in which to draw the mean value.

Used in These Charts
Box Chart

Example:
OutlierSymbol = (NONE, 3, FILLED, green,2, , blue);

Default
ON

typeN the type of symbol to use for outliers in data series N. Legal values are NONE, CIRCLE, SQUARE, DIAMOND, CROSS, TARGET, TRIANGLEDOWN, TRIANGLEUP, IMAGE

sizeN the size in pixels of the symbols for outliers in data series N

styleN the drawing style for outliers in data series N. Legal values are FILLED, OUTLINED or BOTH

borderColorN the color of the border for outliers in data series N

borderWidthN the width in pixels of the border for outliers in data series N

imageN the image to use for displaying for outliers in data series N

colorN the color for outliers in data series N

Attributes
Type, Size, Style, BorderColor, BorderWidth, Image, Color
PercentileN

PercentileN = Integer;

PercentileN defines the value of N when PlotType=PERCENTN. N specifies that for a given data series the chart will show the Nth percentile, median and 100-Nth percentile as a box, and draws whiskers to the minimum and maximum data points.

Used in These Charts
Box

Example:
PercentileN = 5;

PieAngle

PieAngle[N] = Integer;

PieAngle specifies the starting angle for the first pie slice. The angle is measured in degrees from 0 to 360, in a counterclockwise direction, with 0 degrees pointing to the right (i.e., the 3 o'clock position).

Used in These Charts
Pie

Example:
PieAngle = 180;

Attributes
None

PieAngles

PieAngles = Value1, Value2, ...;

PieAngles specifies the starting angle for each pie. The angle is measured in degrees from 0 to 360, in a counterclockwise direction, with 0 degrees pointing to the right (i.e., the 3 o'clock position)

Used in These Charts
MultiPie

Example:
PieAngles 15,0,10,10,....;

Attributes
None
PieBackgrounds

PieBackgrounds = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor),...;

PieBackground specifies the background for each pie.

*Used in These Charts*

MultiPie

*Example:*

PieBackgrounds = (pink,RAISED,10,,,), (lightgray,BOX,1,,);
### Attributes

<table>
<thead>
<tr>
<th>Color</th>
<th>Offsetx</th>
<th>Offsety</th>
<th>Size</th>
</tr>
</thead>
</table>

#### Color

**Color** specifies the base color of the shadow to be drawn behind a chart’s bars.

**Example:**

```plaintext
PieDropShadow = (black_100, "-0.05", "-0.05", 55);
```

#### Offsetx

**Offsetx** specifies the x-coordinate offset from center.

**Example:**

```plaintext
PieDropShadow = (black_100, 25 -10, 25);
```

#### Offsety

**Offsety** specifies the y-coordinate offset from center.

**Example:**

```plaintext
PieDropShadow = (black_100, 25 -10, 25);
```

#### Size

**Size** specifies the width of the blurred area.

**Example:**

```plaintext
PieDropShadow = (black, "-0.05","-0.05", 55);
```

---

### PieEdgeHighlights

**PieEdgeHighlights** = (start,stop,gap,size), ...;

The **PieEdgeHighlights** parameter provides a visual pattern fill in a Pie or MultiPie chart which accents the inner pie border. It overlays a ring (annulus) fill pattern over the existing fill patterns in a
specified zone along the interior edge of the pie. The gap between the sides of the pie and the fill pattern being applied can be modified. The element start sets the beginning color of the gradient, associated with the outside edge; the element stop sets the end color of the gradient, associated with the interior of the pie where the color blends to transparency. Color values are interpolated between the two. The element size specifies the width of the highlight. The element gap specifies the size of the gap between the edge of the highlight and the edge of the pie. When the value for gap is specified as a whole number, it sets the distance between the edge of the highlight and the edge of the pie in pixels. When set to a fractional number between 0 and 1, it sets the gap to a percentage of the radius of the pie. Percentage values are written using a decimal point (e.g. “0.05” and “0.50” represent 5% and 50%, respectively). When using a fractional value, enclose the tuple element in double-quotes to “escape” the decimal point.

**Used in These Charts**
Pie, Multipie

**Example:**
```
PieEdgeHighlights = (blue_25,white_75,1,25), ...;
```

**Attributes**

<table>
<thead>
<tr>
<th>start</th>
<th>stop</th>
<th>gap</th>
<th>size</th>
</tr>
</thead>
<tbody>
<tr>
<td>PieHighlights</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**PieHighlights**

```
PieHighlights = (type,start,stop,angle,gap,extent), ...;
```

The PieHighlights parameter provides a visual pattern fill in a Pie or MultiPie chart. It adds or overlays a fill pattern over one or more existing fill patterns to produce multiple color effects. The angle of origin of the gradient pattern can be modified. The gap between the sides of the pie and the fill pattern being applied can be modified. Gradient patterns can be set using the type attribute. Only gradient patterns may be used. A type value of NONE suppresses the highlights. The element start sets the beginning color of the gradient; the element stop sets the end color of the gradient. Color values are interpolated between the two. The element angle specifies the number of degrees from zero from which the initial gradient color begins at the edge of the pie. The element angle can be set to values greater than 360 degrees. The element gap specifies the size of the gap between the edge of the highlight and the associated edge of the pie. When the value for gap is specified as a whole number, it sets the distance between the edge of the highlight and the edge of the pie in pixels. When set to a fractional number between 0 and 1, it sets the gap to a percentage of the radius of the pie. Percentage values are written using a decimal point (e.g. “0.05” and “0.50” represent 5% and 50%, respectively). When using a fractional value, enclose the tuple element in double-quotes to “escape” the decimal point. The element extent controls the width of the highlight. When the value for extent is specified using a whole number, it sets the diameter of the highlight in pixels. If extent is set to -1, PieHighlights fills the diameter available after taking into account the value of the gap attribute specified previously. If extent is set to a fractional number between 0 and 1, PieHighlights sets the extent of the highlight to that percentage of the diameter of the pie available.
### Gradients

<table>
<thead>
<tr>
<th>Gradients</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRADIENTVERTICAL</td>
<td>bottom to top style gradient</td>
</tr>
<tr>
<td>GRADIENTHORIZONTAL</td>
<td>left to right style gradient</td>
</tr>
<tr>
<td>GRADIENTF DIAG</td>
<td>top right to bottom left style gradient</td>
</tr>
<tr>
<td>GRADIENTBDIAG</td>
<td>top left to bottom right style gradient</td>
</tr>
<tr>
<td>GRADIENTRADIAL</td>
<td>radial style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERHORIZONTAL</td>
<td>center out horizontal style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERVERTICAL</td>
<td>center out vertical style gradient</td>
</tr>
</tbody>
</table>

**start**

This color is used in the following ways:
- **Starting color for gradients**

**stop**

This color is used in the following ways:
- **Stopping color for gradients**

### Used in These Charts

Pie, Multipie

**Example:**

```plaintext
PieHighlights = (GRADIENTRADIAL, yellow, white, 270, 15, -1);
```

### Attributes

<table>
<thead>
<tr>
<th>type</th>
<th>start</th>
<th>stop</th>
</tr>
</thead>
<tbody>
<tr>
<td>angle</td>
<td>gap</td>
<td>extent</td>
</tr>
</tbody>
</table>

### PieLabel

PieLabel = (State, Color, FontName, FontSize, Angle, InteriorAlignment),...;

PieLabel controls the appearance of the text in the pie labels.

**Used in These Charts**

MultPie

**Example:**

```plaintext
PieLabel = ("ON", teal, "Sansserif Bold", 12, 0, null),...;
```

### Attributes

State  Color  FontName  FontSize  Angle  InteriorAlignment
PieLabelBox

PieLabelBox = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor),...

PieLabelBox controls the appearance of the optional box surrounding each pie label.

Used in These Charts
MultiPie

Example:
    PieLabelBox = (teal, "SOLID", 1);

Attributes

Color BorderType BorderWidth ImageURL ImageFormat BorderColor

PieLayout

PieLayout = (Orientation, Row, Columns);

PieLayout defines the layout for the pie series. Orientation is either horizontal or vertical.

Used in These Charts
MultiPie

Example:
    PieLabelLocation = (VERTICAL, 3, 1);

Default
HORIZONTAL

Attributes

Orientation Number of Rows Number Of Columns

PieLabelLocation

PieLabelLocation = Location;

PieLabelLocation defines the location of the pie labels

Location can be set to TOP, BOTTOM, LEFT or RIGHT.

Used in These Charts
MultiPie
Example:

```
PieLabelLocation = LEFT
```

Default

TOP

Attributes

Location

### PieLabels

```plaintext
PieLabels = Label1, Label2, Label3, ... LabelN;
```

Specifies the text for the labels associated with each pie

**Used in These Charts**

MultiPie

**Example:**

```
PieLabels = “North”, “South”...;
```

Default

**Attributes**

Label

### PieMargin

```plaintext
PieMargin = Integer;
```

PieMargin defines the margin between pies in the pie layout.

**Used in These Charts**

MultiPie

**Example:**

```
PieMargin = 1
```
**Attributes**

NONE

---

**PieSize**

```
PieSize = (minWidth, minHeight, maxWidth, maxHeight);
```

The PieSize parameter can be used to set minimum and maximum sizes for the actual pie in a pie chart. This allows programmers to guarantee that the pie portion will always be the same size regardless of the length of the strings in the legend or slice labels. PieSize has the following interaction with PieSquare; if the minimum or maximum dimensions specified are not square, and PieSquare is ON, then the pie will be inscribed in a square with a dimension that ranges from the smallest of the minimum values to the smallest of the maximum values.

**Used in These Charts**

Pie

**Example:**

```
PieSize = (5,25);
```

**Attributes**

`minWidth`, `minHeight`, `maxWidth`, `maxHeight`

---

**PieSpotlights**

```
PieSpotlights = (start, stop, center, centeroffsetx, centeroffsety, focusoffsetx, focusoffsety, radius),...;
```

Adds or overlays a color fill over one or more existing fill patterns to produce multiple layered color effects. The spotlight “illuminates” the pie in the Pie or MultiPie chart. The center of the spotlight and its focus can be adjusted independently by adjusting offsets from the pie chart center point. The elements `centeroffsetx` and `centeroffsety` set the x and y-coordinates of the center of the spotlight as an offset of the pie chart center point. When set to whole numbers, `centeroffsets` and `centeroffsety` specify the number of pixels to offset from the chart center point. When set to fractional values (between 0 and 1), they are interpreted as percentages of the diameter of the pie. Percentage values are written using a decimal point (e.g. “0.05” and “0.50” represent 5% and 50%, respectively). When using a fractional value, enclose the tuple element in double-quotes to “escape” the decimal point. The element `focusoffsetx` is the offset from the pie chart center which defines the x-coordinate of the focus point of the spotlight. The element `focusoffsety` is the offset from the pie chart center which defines the y-coordinate of the focus point of the spotlight. When set to whole numbers, `focusoffsets` and `focusoffsety` specify the offset from the center in pixels. When set to fractional values (between 0 and 1), they are interpreted as percentages of the diameter of the pie. The element `radius` sets the size of the spotlight, from its center to its edge. When set to a whole number, it
sets the size of the \textit{radius} in pixels; when set to a fractional value, it sets the radial diameter of the spotlight based upon that percentage of the diameter of the pie.

\textit{Used in These Charts}

Pie, MultiPie

\textit{Example:}

\begin{verbatim}
PieSpotlights = (blue_0,blue_105,RIGHT,1,1,120,-120,440);
\end{verbatim}

\textit{Attributes}

\begin{verbatim}
start
\end{verbatim}

\begin{verbatim}
stop
\end{verbatim}

\begin{verbatim}
center
\end{verbatim}

\begin{verbatim}
centeroffsetx
\end{verbatim}

\begin{verbatim}
centeroffsety
\end{verbatim}

\begin{verbatim}
focusoffsetx
\end{verbatim}

\begin{verbatim}
radius
\end{verbatim}

\textit{Start}

\begin{verbatim}
Start specifies the first of the two colors which will be interpolated to produce a gradiant spotlight.
\end{verbatim}

\textit{Example:}

\begin{verbatim}
PieSpotlights = (purple_40,blue_155,LEFT,20,10,120,-120,250);
\end{verbatim}

\textit{Stop}

\begin{verbatim}
Stop specifies the second of two colors which will be interpolated to produce a gradiant spotlight.
\end{verbatim}

\textit{Example:}

\begin{verbatim}
PieSpotlights = (purple_40,blue_155,LEFT,20,10,120,-120,250);
\end{verbatim}

\textit{Center}

\begin{verbatim}
Center specifies the position around of the center of the chart where the spotlight center will be placed.
\end{verbatim}

\textit{Example:}

\begin{verbatim}
PieSpotlights = (purple_40,blue_155,LEFT,20,10,120,-120,250);
\end{verbatim}
Values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RIGHT</td>
<td>Offsets the center point of the spotlight to the right of the center point of the chart.</td>
</tr>
<tr>
<td>LEFT</td>
<td>Offsets the center point to the left.</td>
</tr>
<tr>
<td>TOP</td>
<td>Offsets the center point to the top.</td>
</tr>
<tr>
<td>BOTTOM</td>
<td>Offsets the center point to the bottom.</td>
</tr>
<tr>
<td>CENTER</td>
<td>Uses the chart center point for the spotlight center point.</td>
</tr>
<tr>
<td>TOPRIGHT</td>
<td>Offsets the center point of the spotlight to the top right.</td>
</tr>
<tr>
<td>TOPLEFT</td>
<td>Offsets the center point of the spotlight to the top left.</td>
</tr>
<tr>
<td>BOTTOMRIGHT</td>
<td>Offsets the center point of the spotlight to the bottom right.</td>
</tr>
<tr>
<td>BOTTOMLEFT</td>
<td>Offsets the center point of the spotlight to the bottom left.</td>
</tr>
</tbody>
</table>

Default

CENTER

Centeroffsetx

Centeroffsetx specifies the x-coordinate offset for the spotlight center.

Example:

```
PieSpotlights = (purple_40,blue_155,LEFT,20,10,120,-120,250);
```

Centeroffsety

Centeroffsety specifies the y-coordinate offset for the spotlight center.

Example:

```
PieSpotlights = (purple_40,blue_155,LEFT,20,10,120,-120,250);
```

Focusoffsetx

Focusoffsetx specifies the x-coordinate offset for the center of the spotlight’s focus.

Example:

```
PieSpotlights = (purple_40,blue_155,LEFT,20,10,120,-120,250);
```

Focusoffsety

Focusoffsety specifies the y-coordinate offset for the center of the spotlight’s focus.

Example:

```
PieSpotlights = (purple_40,blue_155,LEFT,20,10,120,-120,250);
```
Radius

Radius specifies the length of the radius of the spotlight from the center of the spotlight.

Example:

PieSpotlights = (purple_40, blue_155, LEFT, 20, 10, 120, -120, 250);

PieSquare

PieSquare = Switch;

PieSquare, when turned on, tells the pie chart that the appearance of the pie chart should be kept as high as it is wide. This prevents the pie shape from becoming too wide or too narrow because of titles, labels, legends, or other chart features.

Used in These Charts

Pie MulitPie

Example:

PieSquare = ON;
PieSquare = OFF;

Attributes

None

Pie3Ddepth

Pie3Ddepth[N] = Pixels;

Pie3DDepth specifies the apparent 3D depth of the pie in pixels (whole number).

Used in These Charts

Pie MulitPie

Example:

Pie3DDepth = 12;

Attributes

None
PlotArea

PlotArea = (xlocation, ylocation, width, height);

PlotArea allows chart designers to define and bound the position of the plot area within the chart. The plot area is defined as the area where the data points and axes are displayed. By default, NetCharts charts are laid out with respect to the text elements to the left, right, top, and bottom of the chart while assigning the remaining real estate to the PlotArea. Specifying PlotArea changes this behavior and causes the chart to be laid out with respect to the PlotArea while assigning the remaining real estate to text elements in the top, bottom, left and right of the chart.

If PlotArea is defined in absolute terms (in pixels), the plot area will remain a fixed size, and will be clipped if it does not fit within the chart. If the chart size should change, the extra or eliminated space will affect only the space to the top, bottom, left and right of the plot area. Text in the top, bottom, left or right will be clipped if it does not fit in the space.

If PlotArea is defined in relative terms (as a percentage of the chart size) the PlotArea will grow and shrink proportionally with changes to the chart size. Text in the top, bottom, left or right will be clipped if it does not fit in the space outside the plot area.

- **xlocation** – x coordinate of the upper left corner of the plot area within the chart. If xlocation is a number between 0 and 1, it is interpreted as a percentage of the total available chart width. If xlocation is >= 1, it is interpreted as an absolute location in pixels.

- **ylocation** – y coordinate of the upper left corner of the plot area within the chart. If ylocation is a number between 0 and 1, it is interpreted as a percentage of the total available chart height. If ylocation >= 1 it is interpreted as an absolute location in pixels.

- **width** – width of plot area. If width is a number between 0 and 1, it is interpreted as a percentage of the total available chart area. If width >= 1 it is interpreted as an absolute size in pixels.

- **height** – height of plot area. If height is a number between 0 and 1, it is interpreted as a percentage of the total available chart height. If height >= 1 it is interpreted as an absolute size in pixels.

**Used in These Charts**
Bar, Box, Bubble, Combo, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**

```
PlotArea = (1,1,1,1);
```

**Attributes**

```
xlocation  ylocation  width  height
```
PlotType

PlotType = Type;

PlotType defines the type of plot to be displayed for a BoxChart.

Used in These Charts
Box

Example:
PlotType = STANDARD | EDA | GAUSSIAN | PERCENTN | TENVINETY;

Attributes
Type

Type refers to the manner in which lines are stacked in a line chart, allowing the creation of area graph.

Example:
PlotType = STANDARD;

Values

STANDARD In the Standard type of box plot, the box represents the IQR. The Inter-Quartile Range is the difference between the upper hinge (the 75th percentile) and the lower hinge (the 25th percentile).

EDA The Exploratory Data Analysis (EDA) box chart is similar to the Standard box chart in that the box is based on the IQR, and the median is plotted as a line through the box. (The lower part of the box is also referred to as the lower quartile, and the upper part of the box is then referred to as the upper quartile.) However, this type of plot features additional elements called fences. The fences are defined as follows:
- Lower Outer Fence = lower quartile - 3 * IQR
- Lower Inner Fence = lower quartile - 1.5 * IQR
- Upper Inner Fence = upper quartile + 1.5 * IQR
- Upper Outer Fence = upper quartile + 3 * IQR

GAUSSIAN The Gaussian box chart is different from the other options in terms of which statistics is represents. The lower portion of the box is the minimum value, and the upper portion of the box is the maximum value. The mean value is shown as a line through the box. The lower whisker is equal to the mean value minus three standard deviations, and the upper whisker is equal to the mean value plus three standard deviations.

PERCENTN This represents the minimum, Nth percentile, median, 100-Nth percentile, and the maximum. In this plot, there are no outside values, fences, adjacent values, etc. The minimum value is the end of one whisker, and the maximum value is the end of the other whisker. When N=10 this mode is identical to TENVINETY.
**Common CDL Attributes**

<table>
<thead>
<tr>
<th>Mode</th>
<th>Color</th>
<th>FontName</th>
<th>FontSize</th>
<th>Angle</th>
<th>interiorAlignment</th>
</tr>
</thead>
</table>

**PolarLabel**

PolarLabel = (mode, Color, FontName, FontSize, Angle, interiorAlignment);

PolarLabel defines the presentation format for optional numeric labels for the exterior of a polar chart.

**Used in These Charts**

Polar

**Example:**

PolarAxis = (ON, black, "TimesRoman", 16, 0,LEFT);

**Attributes**

**Mode**

Mode determines whether or not the labels are shown.

**Values**

<table>
<thead>
<tr>
<th>ON</th>
<th>Show tic labels for this axis</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFF</td>
<td>Don’t show tic labels on this axis</td>
</tr>
</tbody>
</table>

**Default**

ON

**interiorAlignment**

interiorAlignment dictates the alignment of text lines in multi-line labels.
**Common CDL Attributes**

<table>
<thead>
<tr>
<th>Values</th>
<th>DataType</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEFT</td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td></td>
</tr>
<tr>
<td>CENTER</td>
<td></td>
</tr>
</tbody>
</table>

**Default**

CENTER

**PolarLabelFormat**

PolarLabelFormat = (dataType,formatString);

The **PolarFormat** parameter defines the format for displaying exterior labels on the polar chart.

**Used in These Charts**

Polar

**Example:**

PolarLabelFormat = (INTEGER, "%d\u00B0");

**Attributes**

**DataType**

FormatString

**Attributes**

**DataType**

The type of number to use when formatting the labels.

**Values**

FLOAT
INTEGER
DECIMAL

**PolarLabelStep**

PolarLabelStep = stepsize;

PolarLabelStep defines the step size for the exterior polar chart labels. For example, a value of 90 would cause labels to be drawn at 0, 90, 180 and 270 degrees around the exterior of the polar chart.

**Used in These Charts**

Polar

**Example:**

PolarLabelStep = 90;
**PolarScale**

\[
\text{PolarScale} = (\text{minValue}, \text{maxValue}, \text{stepSize})\;
\]

The `PolarScale` parameter defines the scales for each of the axes in the polar chart chart. All axes in a polar chart have the same scale. If `PolarScale` is not specified, or only certain attributes are specified, the axes will automatically choose values for the unspecified attributes.

**Used in These Charts**
Polar

**Example:**

```plaintext
PolarScale = (0,10,1);
```

**Attributes**

<table>
<thead>
<tr>
<th>MaxValue</th>
<th>MinValue</th>
<th>StepSize</th>
</tr>
</thead>
</table>

**PolarSquare**

\[
\text{PolarSquare} = \text{ON}|\text{OFF};
\]

The `PolarSquare` parameter tells the polar chart that the data area of the chart should be kept as high as it is wide.

**Used in These Charts**
Polar

**Example:**

```plaintext
RadarSquare = ON;
```

**PolarSize**

\[
\text{PolarSize} = (\text{minWidth}, \text{minHeight}, \text{maxWidth}, \text{maxHeight}) ;
\]

The `PolarSize` parameter can be used to set minimum and maximum sizes for the axes area in a polar chart. This allows programmers to guarantee that the axes area will always be the same size regardless of the length of the strings in the legend or tic labels. `PolarSize` has the following interaction with `PolarSquare`; if the minimum or maximum dimensions specified are not square, and `RadarSquare` is ON, then the axes area will be inscribed in a square with a dimension that ranges from the smallest of the minimum values to the smallest of the maximum values.
**Used in These Charts**

Polar

*Example:*

\[
\text{PolarSize} = (100,100,200,200);
\]

**Attributes**

minWidth, minHeight, maxWidth, maxHeight

---

**PolyActiveLabels**

PolyActiveLabels = ("Label1", "URL1", "Target1"),...;

PolyActiveLabels define the active labels associated with defined polygons.

**Used in These Charts**

Diagram Chart

*Example:*

PolyActiveLabels = ("west region", "javascript:doSelection()");

**Default**

Use tag name as label

**Attributes**

Label, URL, Target

---

**PolyColor**

PolyColor = ("Tag", "Color"),...;

PolyColor associates a color with the named polygon, causing that polygon to be filled with the given color.

**Used in These Charts**

Diagram Chart

*Example:*

PolyColor = ("MD","green_130"),("TX","green_200");

**Default**

None

**Attributes**

TagName, Color
PolySet

PolySet = ("Tag", x1,y1,x2,y2,...);~;

PolySet defines a polygon with a “tag” name and a series of X/Y points.

Used in These Charts

Diagram Chart

Example:

PolySet = ("A",120,120,180,120,180,160,120,160,120,120),
("B",220,220,280,220,280,260,220,260,220,220);

Default

None

Attributes

TagName, X/Y Pairs

RadarSize

RadarSize = (minWidth, minHeight, maxWidth, maxHeight);~;

The RadarSize parameter can be used to set minimum and maximum sizes for the axes area in a radar chart. This allows programmers to guarantee that the axes area will always be the same size regardless of the length of the strings in the legend or tic labels. RadarSize has the following interaction with RadarSquare; if the minimum or maximum dimensions specified are not square, and RadarSquare is ON, then the axes area will be inscribed in a square with a dimension that ranges from the smallest of the minimum values to the smallest of the maximum values.

Used in These Charts

Radar

Example:

RadarSize = (100,100,200,200);

Attributes

minWidth, minHeight, maxWidth, maxHeight

RadarSquare

RadarSquare = ON|OFF;~;
The `RadarSquare` parameter tells the radar chart that the appearance of the `CenterRadius` should be kept as high as it is wide.

**Used in These Charts**
Radar

**Example:**

```plaintext
RadarSquare = ON;
```

### RadialAxes

```
RadialAxes = ("axisTitle", minValue, maxValue, stepSize)...;
```

The `RadialAxes` parameter defines each of the axes in the chart.

**Used in These Charts**
Radar

**Example:**

```plaintext
RadialAxes = RadialAxes = ("Metric1",0,100,25,black),
      ("Metric2",50,100,25,black),("Metric3",0,200,50,black),
      ("Metric4",60,90,10,black),("Metric5"0,95,15,black);
```

**Attributes**

<table>
<thead>
<tr>
<th>AxisTitle</th>
<th>MaxValue</th>
<th>MinValue</th>
<th>StepSize</th>
</tr>
</thead>
</table>

### RadialAxesAngles

```
RadialAxesAngles = angle1, angle2 ...
```

The `RadialAxesColors` parameter controls the drawing angle of the axes on a Polar chart. An angle of 0 degrees draws an axis straight up from the center of the chart.

**Used in These Charts**
Polar

**Example:**

```plaintext
RadialAxesAngles = 0,90,180,270;
```
RadialAxesColors

RadialAxes = color1, color2 ...;

The `RadialAxesColors` parameter defines the colors for each of the axes in the chart.

**Used in These Charts**

Radar, Polar

**Example:**

```
RadialAxes = black, black, red, black;
```

RadialAxesFormat

`RadialAxesFormat = (dataType, formatString),...;`

The `RadialAxesFormat` parameter defines the format for displaying radial axis tic labels.

**Used in These Charts**

Radar, Polar

**Example:**

```
RadialAxesFormat = (INTEGER, "%3d"), (FLOAT, "%f"), (INTEGER, "%3d");
```

**Attributes**

**DataType**

Value that specifies the type of numeric format to use to label the axes tics. Legal values include INTEGER, FLOAT and DECIMAL.

**FormatString**

If the `dataType` attribute is INTEGER or FLOAT, the input data value is expected to be of type integer or float and will be parsed as such (if string conversion is necessary). The format itself is a C-language style `sprintf` format. Some examples:

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Format</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECIMAL, DECIMAL, DECIMAL</td>
<td>&quot;%3d&quot;, &quot;%6.1f&quot;, &quot;%3d&quot;</td>
<td>12, 3.45, 12</td>
</tr>
<tr>
<td>FLOAT, FLOAT, FLOAT</td>
<td>&quot;%4.2f&quot;, &quot;%6.1f&quot;, &quot;%5.0f&quot;</td>
<td>3.14, 1.23, 123</td>
</tr>
<tr>
<td>INTEGER, INTEGER, INTEGER</td>
<td>&quot;%3d&quot;, &quot;%5d&quot;, &quot;%6d&quot;</td>
<td>12, 123, 1234</td>
</tr>
</tbody>
</table>
If the format type is DECIMAL, the format syntax is consistent with those defined in the Java DecimalFormat spec.

**RadialAxesLabel**

RadialAxesLabel = ("Mode", Color, Font, Label), ...;

RadialAxesLabel controls the appearance of all radial axes labels in the chart. One radial axis label may be displayed per axis. The label is drawn at the end of the axis.

**Used in These Charts**

Radar

**Example:**

RadialAxesLabel = ("ON",black,"sansserif",10,0),("ON",red,"sansserif",12, 0);

**Attributes**

<table>
<thead>
<tr>
<th>Font</th>
<th>Color</th>
<th>Label</th>
<th>Mode</th>
</tr>
</thead>
</table>

**RadialAxesLabels**

RadialAxesLabels = ("Axis1Label1",Axis1Label2...), (Axis2Label1,Axis2Label2...)...;

RadialAxesLabels specifies the text to use as axis tic labels for each axis.

**Used in These Charts**

Radar

**Example:**

RadialAxesLabels = ("Jan","Feb","Mar"), ("inbound","outbound","returned");

**RadialAxesScales**

RadialAxesScales = (minValue, maxValue, stepSize) ...;
The **RadialAxesScales** parameter defines the scales for each of the axes in the chart. This parameter was introduced in NetCharts 4.6

**Used in These Charts**
Radar

**Example:**
```
RadialAxesScales = (25,40,5),(0,100,10);
```

**Attributes**
- **MaxValue**
- **MinValue**
- **StepSize**

**RadialAxesTics**

```
RadialAxesTics = ("axisTicLabelMode", "axisTicLabelColor", "axisTicLabelFont", axisTicLabelFontAngle),...;
```

The **RadialAxesTics** parameter specifies the label attributes for the tic marks displayed along the defined axis. The tic labels are generated automatically based on the other parameter settings, and are displayed using the given label attributes in the RadialAxesTics parameter. If any attribute is not defined, any previous value of the attribute will be used.

**Used in These Charts**
Radar, Polar

**Example:**
```
RadialAxesTics = ("ON",black,"sansserif",9,0),("ON",red,"sansserif",12,0);
```

**Attributes**
- **Font**
- **FontAngle**
- **Color**
- **Mode**

**RadialAxesTitles**

```
RadialAxesTitles = “Title1”,“Title2”,“Title3”...;
```

The **RadialAxesTitles** parameter defines the title for each of the axes in the chart. This parameter was introduced in NetCharts 4.6

**Used in These Charts**
Radar

**Example:**
```
RadialAxesTitles = “Inbound”,“Outbound”,“Undelivered”;
```
RadialAxesTitleActiveLabels

RadialAxesTitleActiveLabels = ("Label","URL","Target"), ...;

The RadialAxesTitleActiveLabels parameter specifies a custom active label to be associated with the radial axis title. That is, these labels will be displayed whenever the mouse “dwell”s over the specified radial axis title.

Used in These Charts
Radar

Example:
RadialAxesTitleActiveLabels = ("Metric 1",);

Attributes
Label      Target      URL

RadialGrids

RadialGrids = (gridRadius, gridLineType, gridLineWidth, gridLineColor, gridAreaColor), ...;

The RadialGrids parameter supports the display of one or more circular grids behind the data.

Used in These Charts
Radar, Polar

Example:
RadialGrids = (25,SOLID,1,black,white),(50,SOLID,1,black,white);

Attributes
GridRadius, GridLineType, GridLineWidth, GridLineColor, GridAreaColor

RelativeBoxSymbolWidth

RelativeBoxSymbolWidth = OFF | LINEAR | SQRT;
RelativeBoxSymbolWidth instructs the Box Chart to assign widths to the boxes in the chart based on a function of the relative number of points in each data series. For example, if a value of LINEAR is used, a data series with 100 data points will have a box width twice that of a box representing a data series with only 50 data points. If a value of SQRT is used, the relative box sizes will be calculated as a function of the square root of the number of points in each series.

Used in These Charts

Box Chart

Example:

RelativeBoxSymbolWidth = SQRT;

Default

OFF

Attributes

Mode

RightActiveLabels

RightActiveLabels[N] = ("Label", "URL", "Target"),...;

The right axis labels become active labels when RightActiveLabels parameter is used. Each set in parenthesis has a corresponding set within a DataSet parameter.

Used in These Charts

All

Attributes

Label URL Target

RightAxis

RightAxis[N] = (Label, Color, FontName, FontSize, Angle, interiorAlignment);

If TopAxis is defined for a Combo chart, then the right axis will be used to map the Y data values for all line sets, unless otherwise specified using the LineAxis parameter.

interiorAlignment | Specifies the alignment to use in text strings that contain multiple lines.

The legal values for interiorAlignment are LEFT, RIGHT, or CENTER.
**Common CDL Attributes**

*FormatString*

---

**Used in These Charts**

Combo

**Example:**

RightAxis = ("Milliseconds", black, "TimesRoman", 16, 0);

*Attributes*

Label    Color    FontName    FontSize    Angle    interiorAlignment

---

**RightAxisTitle**

RightAxisTitle[N] = (Label, Color, FontName, FontSize, Angle, interiorAlignment, exteriorAlignment):

The **RightAxisTitle** parameter specifies the label attributes for the axis title, which centered along the top axis, just above the grid. When the **Header** parameter, whether because of the use of a legend, or for some other reason, creates a title that seems visually unbalanced, you may find this parameter produces a more pleasing chart title.

<table>
<thead>
<tr>
<th>interiorAlignment</th>
<th>Specifies the alignment to use in text strings that contain multiple lines.</th>
</tr>
</thead>
<tbody>
<tr>
<td>exteriorAlignment</td>
<td>Specifies the alignment for the entire Title object.</td>
</tr>
</tbody>
</table>

The legal values for **interiorAlignment** and **exteriorAlignment** are **LEFT**, **RIGHT**, or **CENTER**.

**Used in These Charts**

Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**

RightAxisTitle = ("Ceres Prototype Project Schedule\n", black, "Helvetica", 12);

*Attributes*

Label    Color    FontName    FontSize    Angle    interiorAlignment    exteriorAlignment

---

**RightAxisTitleActiveLabel**

RightAxisTitleActiveLabel = ("Label", "URL", "Target");

**RightAxisTitleActiveLabel** defines a single active label destination for the **RightAxisTitle** parameter.

**Used in These Charts**

All
Example:

```
RightAxisTitleActiveLabel = ("Destination", "demo.html", "frame1");
```

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
</table>

**RightAxisTitleBox**

```
RightAxisTitleBox = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);
```

The `RightAxisTitleBox` parameter specifies the region attributes for the axis title centered along the axis. The image-related attributes need not be used unless you want to use an image texture on the box.

**Used in These Charts**
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

Example:

```
RightAxisTitleBox = (lightgray, SHADOW, 3,,gray);
```

**Attributes**

<table>
<thead>
<tr>
<th>Color</th>
<th>BorderType</th>
<th>BorderWidth</th>
<th>ImageURL</th>
<th>ImageFormat</th>
</tr>
</thead>
</table>

**XXCornerStyle**

The drawing style for each of the four corners of the region. Styles are specified in a clockwise fashion starting in the upper left - Top Left, Top Right, Bottom Right, and Bottom Left. Legal values are SQUARE, SNIP and ROUND. The default is SQUARE.

**RightColor**

```
RightColor = Color;
```

`RightColor` controls the color of the top axis and the tic marks, but not the tic mark labels. The default axis color is black. If the NULL color is specified, the axis color is not changed by this parameter.

Example:

```
RightColor = xB5D5F0;
```

**Attributes**

<table>
<thead>
<tr>
<th>Color</th>
</tr>
</thead>
</table>
RightDrawMinorTics

RightDrawMinorTics = ON|OFF;

RightDrawMinorTics controls whether or not right tics are drawn. The default value is ON.

Example:
RightDrawMinorTics = OFF;

Attributes
(Switch)

RightFormat

RightFormat = (FormatType, "FormatExpr", "TimeBase", "TimeUnit");

RightFormat adjusts the numeric labels that are automatically generated for the top axis, should one be defined.

Used in These Charts
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

Example:
RightFormat = (DATE, "%w\n%M/%D", "1 Jan 2000", "1h");
RightFormat = (DATE, "%M/%D", "1 Apr 96", "1d");
RightFormat = (INTEGER);
RightFormat = (FLOAT,"$%,9.2f",,);

Attributes
FormatType    FormatExpr    TimeBase    TimeUnit

FormatType specifies the type of number being displayed on the top axis.

Example:
RightFormat = (DATE, "%w\n%M/%D", "1 Jan 2000", "1h");
RightFormat = (DATE, "%M/%D", "1 Apr 96", "1d");
RightFormat = (INTEGER);
RightFormat = (FLOAT,"$%,9.2f",,);
Common CDL Attributes

Values
DATE Axis value are shown as date and/or time values. See Appendix A: Date and Time Values for further detail.
FLOAT Axis values are shown with decimal parts.
INTEGER Axis values are shown as integers, and are rounded if necessary.

Default
INTEGER

TimeBase

The TimeBase attribute specifies the base date to be used when determining the actual date or time value when using a time unit or numeric value. See Appendix A: Date and Time Values for further detail.

Example:
RightFormat = (DATE, "%w\n%M/%D", "1 Jan 2000", "1h");
RightFormat = (DATE, "%M/%D", "1 Apr 96", "1d");

Values
String values representing dates or times

Default
None

TimeUnit

The TimeUnit attribute controls the time multiplier to be used when determining the actual data/time value when using a numeric value. See Appendix A: Date and Time Values for further detail.

Example:
RightFormat = (DATE, "%w\n%M/%D", "1 Jan 2000", "1h");
RightFormat = (DATE, "%M/%D", "1 Apr 96", "1d");

Values
String values representing dates or times

Default
None
**RightLabels**

RightLabels = "Label1", "Label2", ...;

The RightLabels parameter specifies a list of custom tic mark labels that will be used instead of the numeric labels automatically generated by the axis. The RightLabels will be evenly placed along the axis, overriding any tic placement specified by the StepValue attribute.

**Used in These Charts**

Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**


**Attributes**

RightMargins

RightMargins = (BottomSideMargin, TopSideMargin);

The RightMargins parameter specifies the gap, in pixels, at the beginning and end of the right axis. Most often used to prevent clipping of data points at the extreme ends of the scale.

**Example:**

RightMargins = (20, 20);

**RightScroll**

RightScroll = (ScrollMin, ScrollMax);

The RightScroll parameter specifies a range of values through which an axis can be scrolled. When the ScrollMin and ScrollMax attributes are defined for the axis, the axis will be displayed as a slider bar, using the axis color defined, with a white background. The relative size of the slider represents the percentage of the entire range currently being displayed. That is, it graphically depicts the size of the current axis range (MinValue and MaxValue attributes) relative to the scrollable region (ScrollMin and ScrollMax attributes). See the RightScale parameter for MinValue and MaxValue definitions.

RightScroll should only be used in conjunction with the RightScale parameter.

**Example:**

RightScroll = (0, 98);

**Attributes**

ScrollMin          ScrollMax
**ScrollMin**

*ScrollMin* sets the lower visible limit for a scrollbar defined with *RightScroll*

**Example:**

```
RightScroll = (0, 98);
```

**Values**

<MinValue

**Default**

None

**ScrollMax**

*ScrollMax* sets the upper visible limit for a scrollbar defined with *RightScroll*

**Example:**

```
RightScroll = (0, 98);
```

**Values**

>MaxValue

**Default**

None

**RightTicLength**

```
RightTicLength = Number;
```

The *RightTicLength* parameter defines the size of axis tic marks which are displayed along the right axis of a chart. The parameter will reset the automatically generated tic length. The value defines the number of pixels to use for the length of the tic mark. By default, the number of pixels used is the width of the character zero (0) as found in the font applied to the label. Setting the *RightTicLength* to the value -1 will cause the default size to be used.

**Attributes**

*Number*

**Number**

Apparent length of a right axis tic mark in a chart, in pixels.

**Used in These Charts**

Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

© Visual Mining, Inc 2002-2011
www.visualmining.com
Example:

```
RightTicLength = 10;
```

**Values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No effect (zero length tics are not drawn).</td>
</tr>
<tr>
<td>1 or greater</td>
<td>Whole number length in pixels</td>
</tr>
</tbody>
</table>

**Default**

-1

**RightTics**

```
RightTics = ("Mode", Color, "FontName", FontSize, Angle, interiorAlignment);
```

The `RightTics` parameter specifies the label attributes for the tic marks displayed along the right axis. The tic labels are generated automatically based on the other parameter settings, and are displayed using the given label attributes in the `RightTics` parameter. If any attribute is not defined, any previous value of that attribute will be used.

**Attributes**

<table>
<thead>
<tr>
<th>Mode</th>
<th>Color</th>
<th>FontName</th>
<th>FontSize</th>
<th>Angle</th>
<th>interiorAlignment</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>BackgroundColor</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>rotationPoint</td>
</tr>
</tbody>
</table>

The legal values for `interiorAlignment` are `LEFT`, `RIGHT`, or `CENTER`.

The legal values for `rotationPoint` are `LEFT`, `RIGHT`.

**Used in These Charts**

Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**

```
RightTics = ("OFF", black, "Helvetica", 10);
```

**Mode**

`Mode` determines whether or not the tic labels are shown on that axis.
Example:

RightTics = ("OFF", black, "Helvetica", 10);

Values

ON  Show tic labels for this axis
OFF Don’t show tic labels on this axis

Default

ON

RightTitle

RightTitle  =  ("Label", Color, "FontName", FontSize, Angle, interiorAlignment, exteriorAlignment);

RightTitle describes an optional title, or label, that sits on the left side of a chart, and uses standard attributes for string text, text color, font, font size, and label rotation. As with Header, RightTitle is universally available in Visual Mining chart applications.

Used in These Charts

All, but most useful in Pie, Diagram, and Dial, which do not have axes.

Example:

RightTitle = ("Financial Status", royalblue, Helvetica, 14, 0, LEFT);

<table>
<thead>
<tr>
<th>interiorAlignment</th>
<th>Specifies the alignment to use in text strings that contain multiple lines.</th>
</tr>
</thead>
<tbody>
<tr>
<td>exteriorAlignment</td>
<td>Specifies the alignment for the entire Title object.</td>
</tr>
</tbody>
</table>

The legal values for interiorAlignment and exteriorAlignment are LEFT, RIGHT, or CENTER.

Attributes

Label  Color  FontName  FontSize  Angle  interiorAlignment

RightTitleActiveLabel

RightTitleActiveLabel = ("Label", "URL", "Target");

RightTitleActiveLabel defines a single active label destination for the RightTitle parameter.
**Common CDL Attributes**

**Example:**

```
RightActiveLabel = ("Destination", "demo.html", "framel");
```

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
</table>

**RightTitleBox**

RightTitleBox = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor);

The `RightTitleBox` specifies a background region just for the `RightTitle` parameter.

**Used in These Charts**

All

**Attributes**

<table>
<thead>
<tr>
<th>BorderColor</th>
<th>BorderType</th>
<th>BorderWidth</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageFormat</td>
<td>ImageURL</td>
<td>TRCornerStyle</td>
<td>BRCornerStyle</td>
</tr>
<tr>
<td>BLCornerStyle</td>
<td>CornerColor</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**XXCornerStyle**

The drawing style for each of the four corners of the region. Styles are specified in a clockwise fashion starting in the upper left - Top Left, Top Right, Bottom Right, and Bottom Left. Legal values are SQUARE, SNIP and ROUND. The default is SQUARE.

**Sash**

Sash[N] = (Color, Width, SashMode);

By default, all charts within a multi-chart are allocated a fixed area within the applet based on the `Charts` and `Layout` parameters, with no dividing “sashes.” The `Sash` parameter allows the specification of the color and width of a dividing line between pair of charts similar to that seen in HTML frames.

**Used in These Charts**

Multi-chart
Attributes

Color  Width  SashMode

SashMode

In some situations, it may be desired to allow users to change the relative size of each chart interactively. SashMode switches the ability to move the sashes in Multi-charts.

Example:

<PARAM NAME=Sash VALUE="(xe3e3e3, 4, DYNAMIC)">  

Values

DYNAMIC  Viewer will be allowed to move each sash line in order to change the amount of space allocated to each row/column or chart  
STATIC  Viewer will not be able to move the sash lines

Default

STATIC

Scale

TopScale[N] = mode, ...;  
BottomScale[N] = mode, ...;  
LeftScale[N] = mode, ...;  
RightScale[N] = mode, ...;

The Scale parameter specifies the minimum and maximum data values which will be displayed along the axis. If the Scale parameter is not defined then the tic mark locations will be automatically determined based on the actual data values being displayed. That is, the axis will be “autoscaled” using the current data values to determine “reasonable” values for MinValue, MaxValue and StepValue.

Any combination of MinValue, MaxValue and StepValue may be defined. Those attributes that are not defined will have “reasonable” values chosen for them.

If the StepValue is defined but not as an even multiple of the difference between the MinValue and MaxValue, then no tic mark will be displayed at the Max Value.
ScaleFactor

ScaleFactor = Number;

The number can be -1 or any value, including fractional values, greater than or equal to 0. Specify a scale factor of -1 to have the chart autoscale. Autoscaling causes the chart's text, line widths, borders, and images to grow in proportion to the size of the chart. For instance, if you increased the size of the chart by 2, then the font sizes would all increase by 2.

The ChartWorksDesigner product refers to the ScaleFactor parameter as ChartScaleFactor.

A scale factor of 0 indicates that the chart should disable scaling. All other positive values indicate the percentage of their original size to which the components of the chart (text, line width, borders, etc.) should be scaled. That is, a value of 1.2 indicates that the components of the chart should be 120% their normal size, whereas a value of .3 means that the components should assume a size 30% that of their normal size.

If no ScaleFactor parameter is set in the CDL definition, the chart defaults to scaling disabled (ScaleFactor = 0).
Attributes

*Mode*  
*logBase*

### ScaleSet

- **TopScaleSet[N]** = (Min, Max, Step, Percentage), ...;
- **BottomScaleSet[N]** = (Min, Max, Step, Percentage), ...;
- **LeftScaleSet[N]** = (Min, Max, Step, Percentage), ...
- **RightScaleSet[N]** = (Min, Max, Step, Percentage), ...

The **ScaleSet** defines the minimum and maximum values of a scale and percentage of total space allocated to the axis.

**Example:**

ScaleSet = (Min, Max, Step, Percentage);

### Scroll

- **TopScroll[N]** = (ScrollMin, ScrollMax);
- **BottomScroll[N]** = (ScrollMin, ScrollMax);
- **LeftScroll[N]** = (ScrollMin, ScrollMax);
- **RightScroll[N]** = (ScrollMin, ScrollMax);

The **Scroll** parameter specifies a range of values through which an axis can be scrolled. When the **ScrollMin** and **ScrollMax** attributes are defined for the axis, the axis will be displayed as a slider bar, using the axis color defined, with a white background. The relative size of the slider represents the percentage of the entire range currently being displayed. That is, it graphically depicts the size of the current axis range (MinValue and MaxValue attributes) relative to the scrollable region (ScrollMin and ScrollMax attributes). See the **Scale** parameter for MinValue and MaxValue definitions.

**Scroll** should only be used in conjunction with the **Scale** parameter.

**Example:**

TopScroll = (0, 98);

### Attributes

**ScrollMin**  **ScrollMax**

**ScrollMin** sets the lower visible limit for a scrollbar defined with TopScroll.
**Example:**

```
TopScroll = (0, 98);
```

**Values**

<MinValue

**Default**

None

---

**ScrollMax**

*ScrollMax* sets the upper visible limit for a scrollbar defined with *TopScroll*

**Example:**

```
TopScroll = (0, 98);
```

**Values**

>MaxValue

**Default**

None

---

**SectorActiveLabels[n]**

*SectorActiveLabels*[N] = (*Name*, *Label*, *URL*, *Target*), ...;

*SectorActiveLabels* defines a list of up to 50 active label destinations for named sectors within a dial, in a dial chart. These labels map to the named sectors that were defined using the *Sectors* parameter.

**Used in These Charts**

Dial

**Example:**

```
SectorActiveLabels = ("Danger", ",", "dailysales.html", "infoframe"),
                   ("Warning", ",", "dailysales.html", "infoframe"),
                   ("On Track", ",", "dailysales.html", "infoframe");
```

**Attributes**

<table>
<thead>
<tr>
<th>Name</th>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SectorBorders

SectorBorders = (Name, Type, LineWidth, Color), ...;

SectorBorders controls the appearance of the borders around a named sector within a dial. Note that the fill color for the sector is designated in the Sectors parameter.

Used in These Charts
Dial

Example:
SectorBorders = ("Danger\nZone",NONE,0,null);

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>LineWidth</th>
<th>Color</th>
</tr>
</thead>
</table>

SectorColors

SectorColors = (Name, Color), ...;

SectorColors controls the background color of a sector.

Used in These Charts
Dial

Example:
SectorColors = ("Sector1",green),("Sector2",yellow);

SectorData

SectorData = (Name, StartValue, StopValue), ...;

SectorData controls the point at which named sectors in a dial start and stop, relative to the measurements on the dial.

Used in These Charts
Dial

Example:
SectorData = ("Danger",0,5),
("Warning",5,8),
("On Track",8,10),
("Low",0,3.25),
("Medium",3.25,6.75),
("High",6.75,10);
Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>StartValue</th>
<th>StopValue</th>
</tr>
</thead>
</table>

**StartValue**

StartValue shows the angular place where the sector begins.

*Example:*

```plaintext
SectorData = ("Danger", 0, 5),
             ("Warning", 5, 8),
             ("On Track", 8, 10),
             ("Low", 0.3.25),
             ("Medium", 3.25, 6.75),
             ("High", 6.75, 10);
```

*Values*

Numeric, but depends on how the dial setting are set in the DialScale parameter.

*Default*

None

**StopValue**

StopValue shows the angular place where the sector ends.

*Example:*

```plaintext
SectorData = ("Danger", 0, 5),
             ("Warning", 5, 8),
             ("On Track", 8, 10),
             ("Low", 0, 3.25),
             ("Medium", 3.25, 6.75),
             ("High", 6.75, 10);
```

*Values*

Numeric, but depends on how the dial setting are set in the DialScale parameter.

*Default*

None

**SectorDelete**

```plaintext
SectorDelete = (Name|ALL), ...;
```

*SectorDelete* is used to delete a specific sector, or all sector, in a dial.

*Used in These Charts*

Dial
Example:

SectorDelete = ("Low"), ("Medium");
SectorDelete = ALL;

Attributes
(Switch)

Switch

This switch allows you to either name the sectors that are to be deleted, or to delete all at once.

Example:

SectorDelete = ("Low"), ("Medium");
SectorDelete = ALL;

Values

Name          A string that names a hand
ALL           All the hands

Default
No defaults

SectorDrag

SectorDrag = "ON"|"OFF";

The SectorDrag switch is used to allow or stop the user from dragging the sectors of a dial with the mouse.

Used in These Charts

Dial

Example:

SectorDrag = "ON";
SectorDrag = "OFF";

Attributes
(Switch)

Switch

This switch sets the on/off state.

Example:

SectorDrag = "ON";
SectorDrag = "OFF";
Values

ON  Allows the user to drag the sector around the dial
OFF  Stops the user from dragging the sector around the dial

Default

OFF

SectorEdgeHighlights

\[
\text{SectorEdgeHighlights} = (\text{start}, \text{stop}, \text{gap}, \text{size}), \ldots;
\]

The \text{SectorEdgeHighlights} parameter provides a visual pattern fill in a Dial chart which accents the dial sectors. It overlays a ring (annulus) fill pattern over the existing fill patterns in a specified zone along the interior edge of the sectors. The gap between the sides of the center button and the fill pattern being applied can be modified. The element \text{start} sets the beginning color of the gradient, associated with the outside edge; the element \text{stop} sets the end color of the gradient, associated with the interior of the sectors where the color blends to transparency. Color values are interpolated between the two. The element \text{size} specifies the width of the highlight. The element \text{gap} specifies the size of the gap between the edge of the highlight and the edge of the sectors. When the value for \text{gap} is specified as a whole number, it sets the distance between the edge of the highlight and the edge of the center button in pixels. When set to a fractional number between 0 and 1, it sets the gap to a percentage of the radius of the pie. Percentage values are written using a decimal point (e.g. “0.05” and “0.50” represent 5% and 50%, respectively). When using a fractional value, enclose the tuple element in double-quotes to “escape” the decimal point.

Used in These Charts

Dial

Example:

\[
\text{SectorEdgeHighlights} = (\text{blue}_25, \text{white}_75, 1, 25), \ldots;
\]

Attributes

\[
\begin{array}{cccc}
\text{start} & \text{stop} & \text{gap} & \text{size} \\
\end{array}
\]

SectorFillPattern

\[
\text{SectorFillPattern} = (\text{type}, \text{color1}, \text{color2}, \text{imageURL}), \ldots;
\]

The \text{SectorFillPattern} parameter provides a visual pattern fill for the sector area of a dial chart.

<table>
<thead>
<tr>
<th>Type</th>
<th>NONE</th>
<th>no pattern, do default fill, if any</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Built-In Patterns</td>
<td>Front slash type</td>
</tr>
</tbody>
</table>

CDL Reference Manual – 7.1

© Visual Mining, Inc 2002-2011
www.visualmining.com
### Common CDL Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSLASH</td>
<td>back slash type</td>
</tr>
<tr>
<td>DGRID</td>
<td>diagonal grid lines, (front and back slash lines)</td>
</tr>
<tr>
<td>HORIZONTAL</td>
<td>horizontal lines</td>
</tr>
<tr>
<td>VERTICAL</td>
<td>vertical lines</td>
</tr>
<tr>
<td>GRID</td>
<td>grid lines, (horizontal and vertical lines)</td>
</tr>
</tbody>
</table>

#### Gradients

<table>
<thead>
<tr>
<th>Gradient Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRADIENTVERTICAL</td>
<td>bottom to top style gradient</td>
</tr>
<tr>
<td>GRADIENTHORIZONTAL</td>
<td>left to right style gradient</td>
</tr>
<tr>
<td>GRADIENTFDIAG</td>
<td>top right to bottom left style gradient</td>
</tr>
<tr>
<td>GRADIENTBDIAG</td>
<td>top left to bottom right style gradient</td>
</tr>
<tr>
<td>GRADIENTRADIAL</td>
<td>radial style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERHORIZONTAL</td>
<td>center out horizontal style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERVERTICAL</td>
<td>center out vertical style gradient</td>
</tr>
</tbody>
</table>

#### Images

<table>
<thead>
<tr>
<th>Image Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMAGE</td>
<td>use an image specified in the optional imageUrl element</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Color 1</th>
<th>Uses in the following ways:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foreground color for patterns</td>
<td></td>
</tr>
<tr>
<td>Starting color for gradients</td>
<td></td>
</tr>
<tr>
<td>Ignored in images</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Color 2</th>
<th>Uses in the following ways:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background color for patterns</td>
<td></td>
</tr>
<tr>
<td>Stopping color for gradients</td>
<td></td>
</tr>
<tr>
<td>Ignored in images</td>
<td></td>
</tr>
</tbody>
</table>

| imageUrl | The URL to an image to use as the fill |

### Used in These Charts

Dial

**Example:**

```plaintext
SectorFillPattern = (GRADIENTVERTICAL, blue, white);
```

### Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Color1</th>
<th>Color2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageFormat</td>
<td>ImageURL</td>
<td></td>
</tr>
</tbody>
</table>

### SectorLabels

```plaintext
SectorLabels = (Name, ON|OFF, LabelPos, Color, FontName, FontSize, Angle, interiorAlignment), ...;
```
SectorLabels works identically to the DialTicLabelStyles parameter, and controls the specific appearance and style of the labels for named sectors defined with the Sectors parameter.

<table>
<thead>
<tr>
<th>interiorAlignment</th>
<th>Specifies the alignment to use in text strings that contain multiple lines.</th>
</tr>
</thead>
</table>

The legal values for interiorAlignment are LEFT, RIGHT, or CENTER.

**Used in These Charts**

Dial

**Example:**

```plaintext
SectorLabels = ("Danger","ON",0.6,white,"Helvetica",10,0),
("Warning","ON",0.6,black,"Helvetica",10,0),
("On Track","ON",0.6,white,"Helvetica",10,0),
("Low","ON",1.2,black,"Helvetica",10,0),
("Medium","ON",1.2,black,"Helvetica",10,0),
("High","ON",1.2,black,"Helvetica",10,0);
```

**Attributes**

<table>
<thead>
<tr>
<th>Name</th>
<th>(Switch)</th>
<th>LabelPos</th>
<th>Color</th>
<th>FontName</th>
</tr>
</thead>
<tbody>
<tr>
<td>FontSize</td>
<td>Angle</td>
<td>interiorAlignment</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Switch**

This switch turns the tic labels’ visibility on and off.

**Example:**

```plaintext
DialTicLabelStyles = ("Hour Dial","ON",1.1,black,"Helvetica",14,0),
("Minute Dial","ON",1.1,black,"Helvetica",14,0);
```

**Values**

<table>
<thead>
<tr>
<th>ON</th>
<th>Show the tic labels</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFF</td>
<td>Hide the tic labels</td>
</tr>
</tbody>
</table>

**Default**

None

**LabelPos**

The LabelPos attribute in the dial chart works the same way the LabelPos parameter does for the pie chart: it defines the position of the tic mark labels relative to the width of the dial. A value of 1.1 will place each label just outside the dial face, while a value of 0.6 will place each label inside of the dial face. For instance, clocks are often made with their tic mark labels on the outside of the dial face, but guages tend to have them just on the inside of the dial face along with the tic marks.

**Example:**

```plaintext
DialTicLabelStyles = ("Hour Dial","ON",1.1,black,"Helvetica",14,0),
```
Common CDL Attributes

Values
Positive real numbers, generally between 0 and 2.

Default
None

Sectors

Dial charts can not only have multiple dials, but these dials can be divided up into sectors. We are used to sectors in gauges, where one typically finds green-yellow-red sectors indicating danger or quality levels. The performance scorecard dial chart (see figure below) incorporates them in several ways.

The Sectors parameter identifies and colors the sector within a named dial, and places the sector radially in that dial.

Used in These Charts

Dial

Example:

Sectors = ("Danger", red, "Internal", 80, 20),
("Warning", yellow, "Internal", 80, 20),
("On Track", green, "Internal", 80, 20),
("Low", black, "External", 100, 80),
("Medium", gray, "External", 100, 80),
("High", white, "External"100, 80);

Attributes

Name  Color  DialName  OuterRadius  InnerRadius

DialName

DialName is the name of the dial in which the sector resides.

Example:

Sectors = ("Danger", red, "Internal", 80, 20),
("Warning", yellow, "Internal", 80, 20),
("On Track", green, "Internal", 80, 20),
("Low", black, "External", 100, 80),
("Medium", gray, "External", 100, 80),
("High", white, "External", 100, 80);

Values
Any string value

Default
None
**OuterRadius**

*OuterRadius* controls the outer limit of the sector as a percentage of the dial’s radius.

**Example:**

```plaintext
Sectors = ("Danger", red,"Internal",80,20),
          ("Warning",yellow,"Internal",80,20),
          ("On Track",green,"Internal",80,20),
          ("Low",black,"External",100,80),
          ("Medium",gray,"External",100,80),
          ("High",white,"External",100,80);
```

**Values**

Numerical value from 0 to 100

**Default**

None

**InnerRadius**

*InnerRadius* controls the inner limit of the sector as a percentage of the dial’s radius.

**Example:**

```plaintext
Sectors = ("Danger", red,"Internal",80,20),
          ("Warning",yellow,"Internal",80,20),
          ("On Track",green,"Internal",80,20),
          ("Low",black,"External",100,80),
          ("Medium",gray,"External",100,80),
          ("High",white,"External",100,80);
```

**Values**

Numerical value from 0 to 100

**Default**

None

**SectorLabel**

*SectorLabel* specifies an optional text label to display along with the sector. The label’s appearance is controlled by the *SectorLabels* parameter.

**ShowEightyTwentyLines**

*ShowEightyTwentyLines* = *ON | OFF*;

This parameter defines whether 80/20 lines will be drawn on the Pareto chart. The default value is OFF.
**Common CDL Attributes**

### Style

- **Pareto**

**Example:**

```
ShowEightyTwentyLines = ON
```

### ShowGroupStackLabels

```
ShowGroupStackLabels = ON | OFF;
```

This parameter is used only for grouped stacked barcharts (GraphType=GROUPSTACK). It defines whether stack labels will be drawn at each tic. The default value is OFF.

### ShowDataPoints

```
ShowDataPoints = ON | OFF;
```

ShowDataPoints is used to display the raw data in addition to the summary data for each series. The default is OFF.

**Used in These Charts**

- Box Chart

**Example:**

```
ShowDataPoints = ON;
ShowDataPoints = OFF;
```

**Default**

OFF

**Attributes**

- **Mode**

### SliceAnimationStyle

```
SliceAnimationStyle = GROW | FADE | NONE
```

Specifies how slices initially appear in a pie chart. This parameter is only valid in SVG or SVGWeb output formats.

**Attributes**

- **Style**

**Style**
Style refers to the manner in which lines are first rendered in a line chart.

**Example:**

```plaintext
LineAnimationStyle = BEND;
```

**Values**

- **GROW**: The slices grow and spin their actual value.
- **FADE**: The slices fade in.
- **NONE**: The slices are immediately visible.

**Default**

NONE

**SliceBorder**

`SliceBorder[N] = (LineType, Width, Color);`

In pie charts, this parameter specifies the line style to be used for the border of all pie slices.

**Used in These Charts**

Pie MultiPie

**Example:**

```plaintext
SliceBorder = (DOTTED, 2, blue);
```

**Attributes**

<table>
<thead>
<tr>
<th>LineType</th>
<th>Width</th>
<th>Color</th>
</tr>
</thead>
</table>

**SliceColors**

`SliceColors[N] = Value1, Value2, Value3, Value4, ..., ValueN;`

In pie charts, this parameter specifies the line style to be used for the border of all pie slices.

**Used in These Charts**

Pie

**Example:**

```plaintext
SliceColor = red, white, blue;
```
Attributes
Color

SliceData

SliceData[N] = Value1, Value2, Value3, Value4, ..., Valuen;

SliceData provides a method for specifying pie chart slice data without using the Slices parameter, and consists of a vector of data which may represent either percentages or data values.

Used in These Charts
Pie

Example:
SliceData = 15, 33, 44, 102, 19.45, 88, 47.9;

Attributes
None

SliceFillPattern

SliceFillPattern = (type, color1, color2, imageURL), ...;

The SliceFillPattern parameter provides a visual pattern fill for pie slices in a chart.

<table>
<thead>
<tr>
<th>Type</th>
<th>NONE</th>
<th>no pattern, do default fill, if any</th>
</tr>
</thead>
<tbody>
<tr>
<td>Built-In Patterns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FSLASH</td>
<td>front slash type</td>
<td></td>
</tr>
<tr>
<td>BSLASH</td>
<td>back slash type</td>
<td></td>
</tr>
<tr>
<td>DGRID</td>
<td>diagonal grid lines, (front and back slash lines)</td>
<td></td>
</tr>
<tr>
<td>HORIZONTAL</td>
<td>horizontal lines</td>
<td></td>
</tr>
<tr>
<td>VERTICAL</td>
<td>vertical lines</td>
<td></td>
</tr>
<tr>
<td>GRID</td>
<td>grid lines, (horizontal and vertical lines)</td>
<td></td>
</tr>
<tr>
<td>Gradients</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRADIENTVERTICAL</td>
<td>bottom to top style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTHORIZONTAL</td>
<td>left to right style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTFDIAG</td>
<td>top right to bottom left style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTBDIAG</td>
<td>top left to bottom right style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTRADIAL</td>
<td>radial style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTCENTERHORIZONTAL</td>
<td>center out horizontal style gradient</td>
<td></td>
</tr>
<tr>
<td>GRADIENTCENTERVERTICAL</td>
<td>center out vertical style gradient</td>
<td></td>
</tr>
<tr>
<td>Images</td>
<td></td>
<td></td>
</tr>
<tr>
<td>color1</td>
<td></td>
<td>This color is used in the following ways:</td>
</tr>
<tr>
<td>------------</td>
<td>---</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Foreground color for patterns</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Starting color for gradients</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Ignored in images</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>color 2</th>
<th></th>
<th>This color is used in the following ways:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>- Background color for patterns</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Stopping color for gradients</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Ignored in images</td>
</tr>
</tbody>
</table>

| imageURL   |   | The URL to an image to use as the fill   |

**Used in These Charts**

Pie MulitPie

**Example:**

SliceFillPattern = (GRADIENTVERTICAL,blue,white);

<table>
<thead>
<tr>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>ImageFormat</td>
</tr>
</tbody>
</table>

**SliceFormat**

SliceFormat = (FormatType, "FormatExpr");

In pie charts, SliceFormat affects the numeric labels that are automatically generated for each slice.

**Used in These Charts**

Pie MulitPie

**Example:**

SliceFormat = (FLOAT,"$%,9.2f");

<table>
<thead>
<tr>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>FormatType</td>
</tr>
</tbody>
</table>

**FormatType**

The `FormatType` attribute specifies the type of number being displayed by that axis
**Common CDL Attributes**

**FormatExpr**

The `FormatExpr` attribute specifies a numeric display format to be used for each slice label and each active label generated by default. The format allows a developer to specify numeric formats using expressions similar to those provided in the C/Unix printf function.

**Example:**

```
SliceFormat = (FLOAT,"$%,9.2f");
```

**Values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLOAT</td>
<td>Display numeric values with decimal points</td>
</tr>
<tr>
<td>INTEGER</td>
<td>Display numeric values only as integers, and will round if necessary</td>
</tr>
</tbody>
</table>

**Default**

FLOAT

**SliceLabel**

```
SliceLabel = (State, Color, FontName, FontSize, Angle, interiorAlignment);
```

`SliceLabel` controls the appearance of all the slice labels in a pie chart. This parameter is a standard “tuple.”

<table>
<thead>
<tr>
<th>interiorAlignment</th>
<th>Specifies the alignment to use in text strings that contain multiple lines.</th>
</tr>
</thead>
</table>

The legal values for `interiorAlignment` are LEFT, RIGHT, or CENTER.

**Used in These Charts**

Pie MultiPie

**Example:**

```
SliceLabel = (ON, Black, Helvetica, 12, 270,LEFT);
```
Common CDL Attributes

Attributes

State  Color  FontName  FontSize  Angle  interiorAlignment

State

State indicates if this parameter is in effect or not.

Example:

SliceLabel = (ON, Black, Helvetica, 12, 270);

Values

ON or null

Default

ON

SliceLabelBox

SliceLabelBox = (Color, BorderType, BorderWidth);

SliceLabelBox controls the appearance of the area under the slice label in a pie chart.

Used in These Charts

Pie MultiPie

Example:

SliceLabelBox = (fuschia, RECESS, 4);

Attributes

Color  BorderType  BorderWidth

SliceLabelContent

SliceLabelContent = [Value1],[Value2],[Value3];

SliceLabelContent allows you to specify the contents of RADIAL or EXTERIOR pie labels. (LEGEND labels will only show the text label.) The SliceLabelContent is a comma-delimited list of up to three items that should appear in these labels.

Used in These Charts

Pie MultiPie
Common CDL Attributes

Example:

SliceLabelContent = DATA,LABEL;

Values

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PERCENTAGE</td>
<td>Displays the slice percentage as nn.nn% (for backward compatibility)</td>
</tr>
<tr>
<td>PERCENTAGE_FLOAT</td>
<td>Displays the slice percentage as nn.nn%</td>
</tr>
<tr>
<td>PERCENTAGE_INT</td>
<td>Displays the slice percentage as nn%</td>
</tr>
<tr>
<td>DATA</td>
<td>The data value, as formatted in the SliceFormat parameter</td>
</tr>
<tr>
<td>LABEL</td>
<td>The label passed as part of the Slices or SliceLabels parameter</td>
</tr>
</tbody>
</table>

Default

LABEL

SliceLabelContentDelimiter

SliceLabelContentDelimiter = "delimiter";

SliceLabelContentDelimiter allows specification of the delimiter to use between content items. These delimiters can be any text, but will usually be commas, newlines (\n), or spaces.

Used in These Charts

Pie MulitPie

Example:

SliceContentDelimiter = ",";
SliceContentDelimiter = "rrr";
SliceContentDelimiter = "\n";
SliceContentDelimiter = " ";

Attributes

None

SliceLabelLine

SliceLabelLine = (LineStyle, LineWidth, Color);

SliceLabelLine controls the appearance of the lines connecting the exterior labels to the pie slice in a pie chart. If the color of the line is set to NULL, or left undefined, the color of each line will be the same as the slice to which it connects.

Used in These Charts

Pie MulitPie
Example:

\[
\text{SliceLabelBox} = (\text{SOLID}, 2, \text{antiquewhite});
\]

Attributes

- LineStyle
- LineWidth
- Color

SliceLabels

\[
\text{SliceLabels} = \text{Label1}, \ \text{Label2}, \ \text{Label3}, \ \text{Label4}, \ldots, \ \text{Labeln};
\]

SliceLabels provides a method for specifying pie chart slice labels without using the Slices parameter, and consists of a vector of labels that will be applied to each of the pie chart slices.

Used in These Charts

- Pie
- MultiPie

Example:

\[
\text{SliceLabels} = \text{"Monday"}, \ \text{"Tuesday"}, \ \text{"Wednesday"}, \ \text{"Thursday"}, \ \text{"Friday"};
\]

Attributes

- Label

SliceLabelStyle

\[
\text{SliceLabelStyle} = \text{Style};
\]

SliceLabelStyle allows explicit specification of the label style for pie charts.

Used in These Charts

- Pie
- MultiPie

Example:

\[
\text{SliceLabelStyle} = \text{EXTERIOR};
\text{SliceLabelStyle} = \text{LEGEND};
\text{SliceLabelStyle} = \text{RADIAL};
\]

Attributes

- Style

Style

Only one Style attribute is allowed for SliceLabelStyle.
**Example:**

```
SliceLabelStyle = EXTERIOR;
SliceLabelStyle = LEGEND;
SliceLabelStyle = RADIAL;
```

**Values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXTERIOR</td>
<td>Labels are put on the left and right sides of the pie.</td>
</tr>
<tr>
<td>LEGEND</td>
<td>Labels are placed in a Legend</td>
</tr>
<tr>
<td>RADIAL</td>
<td>Labels are put along the radius of each slice</td>
</tr>
</tbody>
</table>

If `SliceLabelStyle` isn’t defined, but a Legend is, the style will be LEGEND.

If the chart doesn’t have `SliceLabelStyle`, but `LabelPos` is defined, the style will be RADIAL.

If no `SliceLabelStyle` is defined, but both `LabelPos` and Legend are, then the style shown will be LEGEND.

**Default**

EXTERIOR

---

**SlicePos**

```
SlicePos[N] = Position1, Position2, Position3, ...;
```

`SlicePos` defines the position of each pie slice relative to the width of the pie. Can be used to “pre-explode” slices from the rest of the pie.

**Used in These Charts**

Pie MultiPie

**Example:**

```
SlicePos = 0, 1.2, 1.5, 0, 0, 0, 0;
```

**Attributes**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Position</td>
<td></td>
</tr>
</tbody>
</table>

**Position**

The `Position` attribute may be expressed as a fractional number in the range 0.0 to 2.0, or as a percentage value in the range of 2 to 100. A value of 0.5 or 50.0 will move the slice center radially out 50 percent of the width of the pie. That is, any nonzero value will result in the slice being exploded.

**Example:**

```
SlicePos = 0, 1.2, 1.5, 2.0, 0, 0, 0;
SlicePos = 0, 80, 75, 100, 0, 0, 0;
```
### Common CDL Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Values</td>
<td>Slices all lie together in the pie</td>
</tr>
<tr>
<td></td>
<td>Slices explode from pie</td>
</tr>
<tr>
<td></td>
<td>% amount slices explode from pie</td>
</tr>
<tr>
<td>Default</td>
<td>0</td>
</tr>
</tbody>
</table>

### Slices

Slices[N] = (Value, SliceColor, Label, LabelColor, FontName, FontSize, LabelAngle, LabelBgColor, LabelBgBorder), ...;

Slices has been deprecated. Use SliceData, SliceLabels and SliceColors.

Slices defines the set of slices for a pie chart, specifically the value and appearances for a slice. Each grouping of attributes addresses one of the pie’s slices. You can define each slice individually, or slices will take their attributes from the last grouping defined before them.

Only the first two attributes, Value and SliceColor, need to be specified; all others will be assigned default values. For the first slice, the defaults will be based on system defaults. For all other slices, the value assigned to the previous slice for that attribute will be used as the default. In that way, you need only specify attributes for the first slice in order to control the attributes of all slices. The value NULL may be passed as a pie slice value, but has the same effect as a 0 slice value.

### Used in These Charts

Pie

**Example:**

Slices = (2300, NULL, "Mon", black, "TimesRoman", 14),
(3700,"Tue"),
(1200,"Wed"),
(2500,"Thu"),
(4300,"Fri"),
(1900,"Sat"),
(2700,"Sun");

### Attributes

<table>
<thead>
<tr>
<th>Value</th>
<th>SliceColor</th>
<th>Label</th>
<th>LabelColor</th>
</tr>
</thead>
<tbody>
<tr>
<td>FontName</td>
<td>FontSize</td>
<td>LabelAngle</td>
<td>LabelBgColor</td>
</tr>
<tr>
<td>LabelBgBorder</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Value

Value is the raw numeric data associated with the pie slice described in its group. Visual Mining pie charts automatically calculate percentages, so you do not need to figure this out for yourself. The percentage will be computed based on the total of all slice values. Using NULL as a Value creates a slice of zero value, and is equivalent to a 0 data value.
Common CDL Attributes

SliceColor

Example:

Slices = (2300, NULL, "Mon", black, "TimesRoman", 14),
(3700,"Tue"),
(1200,"Wed"),
(2500,"Thu"),
(4300,"Fri"),
(1900,"Sat"),
(2700,"Sun");

Values
Any real number.

Default
None

SliceColor

determines the color of the pie slice, and is in all respects specified as any other Color attribute. If NULL is used as a value, the color is taken from either a specified ColorTable parameter, or the default system color table.

Example:

Slices = (2300, NULL, "Mon", black, "TimesRoman", 14),
(3700,"Tue"),
(1200,"Wed"),
(2500,"Thu"),
(4300,"Fri"),
(1900,"Sat"),
(2700,"Sun");

Slices = (2300, x6C5D94, "Mon", x6C5D94, "Helvetica", 12, 0, white, NONE, 0),
(3700, x999966, "Tue", x999966, "Helvetica", 12, 0, white, NONE, 0),
(1200, x315394, "Wed", x315394, "Helvetica", 12, 0, white, NONE, 0),
(2500, x213321, "Thu", x213321, "Helvetica", 12, 0, white, NONE, 0),
(4300, x00566F, "Fri", x00566F, "Helvetica", 12, 0, white, NONE, 0),
(1900, x690931, "Sat", x690931, "Helvetica", 12, 0, xe3e3e3, NONE, 0),
(2700, x515F23, "Sun", x515F23, "Helvetica", 12, 0, xe3e3e3, NONE, 0);

Values
See the Color attribute in Chapter 4: Common CDL Attributes for details.

Default
blue

LabelColor

LabelColor determines the color of the pie slice’s label font, and is in all respects specified as any other Color attribute.

Example:

Slices = (2300, NULL, "Mon", black, "TimesRoman", 14),
(3700,"Tue"),
(1200,"Wed"),
(2500,"Thu"),
Common CDL Attributes

LabelAngle

LabelAngle determines the display angle of the pie slice’s label font. In all other respects, it is identical with the common Angle attribute.

Example:

Slices = (2300, x6C5D94, "Mon", x6C5D94, "Helvetica", 12, 0, white, NONE, 0),
(3700, x999966, "Tue", x999966, "Helvetica", 12, 90, white, NONE, 0),
(1200, x315394, "Wed", x315394, "Helvetica", 12, 90, white, NONE, 0),
(2500, x213321, "Thu", x213321, "Helvetica", 12, 90, white, NONE, 0),
(4300, x00566F, "Fri", x00566F, "Helvetica", 12, 270, white, NONE, 0),
(1900, x690931, "Sat", x690931, "Helvetica", 12, 270, gray, NONE, 0),
(2700, x515F23, "Sun", x515F23, "Helvetica", 12, 0, x3e3e3, NONE, 0);

Values

0 degrees counterclockwise from horizontal
90 degrees counterclockwise from horizontal
180 degrees counterclockwise from horizontal
270 degrees counterclockwise from horizontal

Default

0

LabelBgColor

LabelBgColor determines the color behind the pie slice’s label font, and is in all respects specified as any other Color attribute.

Example:

Slices = (2300, x6C5D94, "Mon", x6C5D94, "Helvetica", 12, 0, white, NONE, 0),
(3700, x999966, "Tue", x999966, "Helvetica", 12, 0, white, NONE, 0),
(1200, x315394, "Wed", x315394, "Helvetica", 12, 0, white, NONE, 0),
(2500, x213321, "Thu", x213321, "Helvetica", 12, 0, white, NONE, 0),
(4300, x00566F, "Fri", x00566F, "Helvetica", 12, 0, white, NONE, 0),
(1900, x690931, "Sat", x690931, "Helvetica", 12, 0, x3e3e3, NONE, 0),
Common CDL Attributes

LabelBgBorder

(2700, x515F23, "Sun", x515F23, "Helvetica", 12, 0, xe3e3e3, NONE, 0);

Values

See the Color attribute in Chapter 4: Common CDL Attributes for details.

Default

blue

LabelBgBorder

LabelBgBorder determines the width of the border of the pie slice's label, and is in all respects specified as any other common BorderType attribute.

Example:

Slices = (2300, x6C5D94, "Mon", x6C5D94, "Helvetica", 12, 0, white, NONE, 0),
(3700, x999966, "Tue", x999966, "Helvetica", 12, 0, white, NONE, 0),
(1200, x315394, "Wed", x315394, "Helvetica", 12, 0, white, NONE, 0),
(2500, x213321, "Thu", x213321, "Helvetica", 12, 0, white, NONE, 0),
(4300, x00566F, "Fri", x00566F, "Helvetica", 12, 0, white, NONE, 0),
(1900, x690931, "Sat", x690931, "Helvetica", 12, 0, xe3e3e3, RAISED, 0),
(2700, x515F23, "Sun", x515F23, "Helvetica", 12, 0, xe3e3e3, RAISED, 0);

Values

NONE No border
BOX Simple Box outline
SHADOW Shadow border
RAISED Raised border
RECESS Recessed border

Default

NONE

SliceSet

SliceSet = Value1, Value2, Value3, Value4, ..., Valuen;

SliceSet defines the numeric values for each slice set.

Used in These Charts

MultiPie

SliceSets

SliceSets = ("Name", Color, "State");

SliceSet defines a list of SliceSet tuples with the following attributes in each tuple:
Common CDL Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color</td>
<td>Slice Color.</td>
</tr>
<tr>
<td>State</td>
<td>Labels ON or OFF. Default is ON</td>
</tr>
</tbody>
</table>

**Used in These Charts**

MultiPie

**Example:**

```plaintext
SliceSets = ("Over 18", "x7996A1"), ("Under 18", "xADD6E6");
```

**Attributes**

<table>
<thead>
<tr>
<th>Name</th>
<th>Color</th>
<th>State</th>
</tr>
</thead>
</table>

**SliceSlide**

```plaintext
SliceSlide = ON| OFF;
```

When `SliceSlide = ON`, clicking on a pie slice will cause the slice to slide in/out of the pie. This parameter is only valid in SVG or SVGweb output formats.

**Example:**

```plaintext
SliceSlide = ON;
```

**Attributes**

(Switch)

**StackDisplayOrder**

```plaintext
StackDisplayOrder = BOTTUMUP|TOPDOWN;
```

`StackDisplayOrder` defines the ordering of legend items in a chart with multiple bar series. The default is `BOTTOMUP`, which specifies that the legend items will be displayed in the order in which the data sets are specified. For example, `DataSet1` will appear in the legend first. `TOPDOWN` is useful when multiple series of bars are stacked and the legend is displayed vertically. In this mode, the stack of legend items will be "stacked" in the same order as the bars in the chart.

**Used in These Charts**

Bar, Combo, Pareto, Stock
**Example:**

```
StackDisplayOrder = TOPDOWN;
```

**StackedBarConnectors**

```
StackedBarConnectors = OFF | LINE | FILL;
```

`StackedBarConnectors` defines the bars in a particular series can be connected together with lines. This improves the user’s ability to track the values of a single series of values across a multiseries stacked barchart. This is valid on for multiseries barcharts or combocharts when `GraphType` is `STACK`. The lines are drawn using the drawing attributes specified in the `BarBorder` parameter. If `BarBorder` not defined, one pixel solid black lines are drawn.

OFF specifies that no connectors are drawn.

LINE specifies that lines are drawn between the top and bottom of bars in the same series.

FILL specifies that lines are drawn between the top and bottom of bars in the same series and the area between those lines is filled with the bar set color.

**Used in These Charts**

Bar, Combo

**StackLabel**

```
StackLabel[N] = Type;
```

In the context of stacked bar or line charts, `StackLabel` defines how the default active labels should be generated for each line symbol when `GraphType` is `STACK` or `PERCENT`.

**Used in These Charts**

Bar, Combo, Line, Pareto, Stock

**Example:**

```
StackLabel = TOTAL;
```

**Attributes**

Type

**Type**

Type, in context of the `StackLabel` parameter, determines how the numeric value of the active label is shown.
Example:

\[
\text{StockLabel} = \text{TOTAL};
\]

Values

- **TOTAL**: Uses the accumulated numeric total for the label
- **ITEM**: Uses the individual item’s numeric value for the label
- **PERCENT**: Uses the accumulated numeric total for the label, displayed as a percentage of the total of all values.

Default

TOTAL

StockAnimationStyle

\[
\text{StockAnimationStyle} = \text{GROW} \mid \text{FADE} \mid \text{NONE}
\]

Specifies how the stock series (high, low, open, close) initially appear in a Stock chart. This parameter is only valid in SVG or SVGWeb output formats.

Attributes

Style

Style

Style refers to the manner in which stock data series are first rendered in a stock chart.

Example:

\[
\text{StockAnimationStyle} = \text{GROW};
\]

Values

- **GROW**: The stock series data start as a straight line at 0 and each point bends to its actual value.
- **FADE**: The stock series fade in.
- **NONE**: The series are immediately visible.

Default

NONE

StockAxis

\[
\text{StockAxis}[N] = (X\text{Axis}_1, Y\text{Axis}_1), (X\text{Axis}_2, Y\text{Axis}_2), \ldots;
\]

StockAxis defines which side to be used when mapping the X and Y axes, respectively.
Common CDL Attributes

XAxis

**Used in These Charts**

Stock

**Example:**

```
StockAxis = (BOTTOM, LEFT), (RIGHT, TOP);
```

**Attributes**

XAxis    YAxis

**XAxis**

XAxis, in context of the StockAxis parameter, determines whether the X-axis for a pair of axes will be on the bottom or the top of the chart.

**Example:**

```
StockAxis = (TOP, LEFT), (BOTTOM, RIGHT);
```

**Values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOTTOM</td>
<td>X values will be plotted along the chart’s bottom axis</td>
</tr>
<tr>
<td>TOP</td>
<td>X values will be plotted along the chart’s top axis</td>
</tr>
</tbody>
</table>

**Default**

BOTTOM

**YAxis**

YXAxis, in context of the StockAxis parameter, determines whether the Y-axis for a pair of axes will be on the left or the right of the chart.

**Example:**

```
StockAxis = (TOP, LEFT), (BOTTOM, RIGHT);
```

**Values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEFT</td>
<td>X values will be plotted along the chart’s left axis</td>
</tr>
<tr>
<td>RIGHT</td>
<td>X values will be plotted along the chart’s right axis</td>
</tr>
</tbody>
</table>

**Default**

LEFT

**StockColorTable**

```
StockColorTable[1-50] = Color1, Color2, Color3, Color4, Color5, ...;
```

StockColorTable defines a set of colors for dataset N that overrides all other color specifications for that set. The parameters used for specifying the color of data points in a chart are (in ascending order of precedence) ColorTable, StockSets, StockFillPattern and StockColorTable. StockColorTable is used most frequently to color some specific stock data point.
For example

```
StockColorTable2 = ,,blue;
```

will change the third stock data point in the second series to blue, while all other data points in the chart continue to be colored by one of the other color related parameters.

The colors you can use are defined in the common color attribute (Chapter 4).

**Used in These Charts**

**Stock**

*Example:*

```
StockColorTable2 = ,, red;
```

**Attributes**

None

**StockData[n]**

```
StockData[1-50] = (High1, Low1, Open1, Close1), (High2, Low2, Open2, Close2), ...;
```

*StockData* is used to define stock values.

**Used in These Charts**

**Stock**

*Example:*

```
StockData = (120,119,120,121), (32,30.25, 29.75, 31.5);
```

**Attributes**

<table>
<thead>
<tr>
<th>High</th>
<th>Low</th>
<th>Open</th>
<th>Close</th>
</tr>
</thead>
</table>

**High**

*High* defines the highest stock value for that data point.

*Example:*

```
StockData = (120,119,120,121), (32,30.25, 29.75, 31.5);
```

**Values**

- Any stock number
- null: No symbol will be displayed for this data point

**Default**

none

**Low**
**Low** defines the lowest stock value for that data point.

*Example:*

\[
\text{StockData} = (120, 119, 120, 121), (32, 30.25, 29.75, 31.5);
\]

**Values**

- Any stock number
- **null** The value of **High** will be used in place of **Low**

**Default**

- none

---

**Open**

**Open** defines the opening stock value for that data point.

*Example:*

\[
\text{StockData} = (120, 119, 120, 121), (32, 30.25, 29.75, 31.5);
\]

**Values**

- Any stock number
- **null** No tic mark will be drawn for this data point

**Default**

- none

---

**Close**

**Close** defines the closing stock value for that data point.

*Example:*

\[
\text{StockData} = (120, 119, 120, 121), (32, 30.25, 29.75, 31.5);
\]

**Values**

- Any stock number
- **null** No tic mark will be drawn for this data point

**Default**

- none

---

**StockFillPattern**

\[
\text{StockFillPattern}[N] = (\text{type}, \text{color1}, \text{color2}, \text{imageURL}), ...;
\]

The **StockFillPattern** parameter provides a visual pattern fill for stock sets in a stock chart.
### Common CDL Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Built-In Patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>no pattern, do default fill, if any</td>
</tr>
<tr>
<td>FS\slash</td>
<td>front slash type</td>
</tr>
<tr>
<td>BS\slash</td>
<td>back slash type</td>
</tr>
<tr>
<td>DGRID</td>
<td>diagonal grid lines, (front and back slash lines)</td>
</tr>
<tr>
<td>HORIZONTAL</td>
<td>horizontal lines</td>
</tr>
<tr>
<td>VERTICAL</td>
<td>vertical lines</td>
</tr>
<tr>
<td>GRID</td>
<td>grid lines, (horizontal and vertical lines)</td>
</tr>
</tbody>
</table>

#### Gradients

<table>
<thead>
<tr>
<th>Gradient</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRADIENTVERTICAL</td>
<td>bottom to top style gradient</td>
</tr>
<tr>
<td>GRADIENTHORIZONTAL</td>
<td>left to right style gradient</td>
</tr>
<tr>
<td>GRADIENTFDIAG</td>
<td>top right to bottom left style gradient</td>
</tr>
<tr>
<td>GRADIENTBDIAG</td>
<td>top left to bottom right style gradient</td>
</tr>
<tr>
<td>GRADIENTRADIAL</td>
<td>radial style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERHORIZONTAL</td>
<td>center out horizontal style gradient</td>
</tr>
<tr>
<td>GRADIENTCENTERVERTICAL</td>
<td>center out vertical style gradient</td>
</tr>
</tbody>
</table>

#### Images

<table>
<thead>
<tr>
<th>Image Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMAGE</td>
<td>use an image specified in the optional imageURL element</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>color1</th>
<th>This color is used in the following ways:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- Foreground color for patterns</td>
</tr>
<tr>
<td></td>
<td>- Starting color for gradients</td>
</tr>
<tr>
<td></td>
<td>- Ignored in images</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>color2</th>
<th>This color is used in the following ways:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- Background color for patterns</td>
</tr>
<tr>
<td></td>
<td>- Stopping color for gradients</td>
</tr>
<tr>
<td></td>
<td>- Ignored in images</td>
</tr>
</tbody>
</table>

| imageURL      | The URL to an image to use as the fill            |

---

**Used in These Charts**

Stock

**Example:**

```plaintext
StockFillPattern = (DGRID, blue, white);
```

**Attributes**

<table>
<thead>
<tr>
<th>Type</th>
<th>Color1</th>
<th>Color2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageFormat</td>
<td>ImageURL</td>
<td></td>
</tr>
</tbody>
</table>

---

© Visual Mining, Inc 2002-2011
www.visualmining.com
Common CDL Attributes

<table>
<thead>
<tr>
<th>Width</th>
</tr>
</thead>
</table>

StockLabels[n]

StockLabels[1-50] = ("Label", "URL", "Target"), ...

StockLabels can be used to override the default active labels generated for each data value and/or to specify a hyper-link to another document, in a given target window or frame.

Used in These Charts
Stock

Example:

StockLabels1 = ("DCX", "http://www.daimlerchrysler.com", "infoframe"),
("MSFT", "http://www.microsoft.com", "infoframe"),

Attributes
Label       URL      Target

StockSets

StockSets[N] = (Label1, Color1, Width1, TicLen1), (Label2, Color2, Width2, TicLen2), ...

StockSets defines the display attributes for one or more stock data sets.

Used in These Charts
Stock

Example:

StockSets = ("DCX", azure, 1, 3),
("MSFT", darkblue, 1, 3),
("SEBL", moccasin, 1,3);

Attributes
Label       Color      Width     TicLen

Width

Width defines the width of the vertical bar (hi/lo) in pixels. These pixel values override any StockWidth attributes for the current stock set.

Example:

StockSets = ("DCX", azure, 1, 3),
("MSFT", darkblue, 1, 3),
("SEBL", moccasin, 1,3);
Common CDL Attributes

TicLen

TicLen defines the length of the open/close tic marks in pixels. These pixel values override any StockWidth attributes for the current stock set.

Example:

```plaintext
StockSets = ("DCX", azure, 1, 3),
            ("MSFT", darkblue, 1, 4),
            ("SEBL", moccasin, 1, 5);
```

Values
Any pixel value

Default
0

StockWidth

StockWidth[N] = (Width, TicLen);

StockWidth specifies the relative size of every bar and tic mark used when displaying stock values.

Used in These Charts
Stock

Example:

```plaintext
StockWidth = (5, 80);
StockWidth = (.2, .5);
```

Attributes
Width       TicLen

Width

Width controls the width of the bars for stock data points.
Example:

```
StockWidth = (5, 80);
StockWidth = (.2, .5);
```

Values

0 to 100 or 0.0 to 1.0

- Width of bar in percentage of space allocated for the stock symbol
- The bar will occupy all of the space allocated for each stock symbol, leaving no room for the tic marks

Default

33

**TicLen**

TicLen controls the width of the tic marks indicating open & close values for stock data points.

Example:

```
StockWidth = (5, 80);
StockWidth = (.2, .5);
```

Values

0 to 100 or 0.0 to 1.0

- Width of bar in percentage of space allocated for the stock symbol
- The tic mark will fill up the remaining space not used by the bar

Default

0

**StripLayout**

```
StripLayout[N] = (NumSlots, InitialFill, MaxFill, UndefinedString);
```

StripLayout defines basic strip chart information, including how many "steps" there are in the chart's X axis; which side of the chart the data should begin fill from; the maximum number of datapoints that can be simultaneously loaded without loss; and the string used when a tic is shown but no data for the tic is available.

**Used in These Charts**

Strip

Example:

```
StripLayout = (30, RIGHT, 45, "none");
```
**Attributes**

<table>
<thead>
<tr>
<th>NumSlots</th>
<th>InitialFill</th>
<th>MaxFill</th>
<th>UndefinedString</th>
</tr>
</thead>
</table>

**NumSlots**

NumSlots controls how many "steps" there are in the chart's X axis.

*Example:*

```plaintext
StripLayout = (30, RIGHT, 45, "none");
```

**Values**

Whole number greater than 0

**Default**

None

**InitialFill**

InitialFill controls which side of the chart the data begins to fill in from.

*Example:*

```plaintext
StripLayout = (21, RIGHT, 20, "*");
StripLayout = (22, LEFT, 100, "*");
```

**Values**

LEFT

RIGHT

**Default**

RIGHT

**MaxFill**

MaxFill controls the maximum number of datapoints that can be simultaneously loaded without loss.

*Example:*

```plaintext
StripLayout = (22, RIGHT, 100, "*");
```

**Values**

Whole number

**Default**

None
**UndefinedString**

UndefinedString controls the string used when a tic is shown but no data for the tic is available.

*Example:*

```
StripLayout = (8, RIGHT, 100, "*");
```

**Values**

String value, generally a single symbol.

**Default**

None

**TaskColorTable**

```
TaskColorTable[1-50] = Color1, Color2, Color3, Color4, Color5, ...;
```

TaskColorTable defines a set of colors for dataset N that overrides all other color specifications for that set. The parameters used for specifying the color of tasks in a chart are (in ascending order of precedence) ColorTable, DataSets, and TaskColorTable. TaskColorTable is used most frequently to color some specific task.

For example

```
TaskColorTable2 = ,,blue;
```

will change the third task in the second series to blue, while all other tasks in the chart continue to be colored by one of the other color related parameters.

The colors you can use are defined in the common Color attribute (Chapter 4).

**Used in These Charts**

Time

*Example:*

```
TaskColorTable2 = ,, red;
```

**Attributes**

None

**TaskHeight**

```
TaskHeight = value;
```

The TaskHeight parameter specifies the height of a task bar in a timechart.

**Used in These Charts**

Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y
**Example:**

```
```

**Attributes**

- **Label**

---

**TicLocations**

```
TopTicLocations[N] = value, value, value, ...;
BottomTicLocations[N] = value, value, value, ...;
LeftTicLocations[N] = value, value, value, ...;
RightTicLocations[N] = value, value, value, ...;
```

The **TicLocations** parameter can be used with a corresponding **Label** parameter for complete control of tic drawing and tic labeling. The labels specified in the **Labels** parameter are drawn in order at the locations specified in this parameter. For best results, this parameter should be used in conjunction with an explicitly set axis scale.

**Used in These Charts**

**Example:**

```
TopTicLocations = 10, 20, 30, 40;
```

**Attributes**

- **value**

---

**Tics**

```
TopTics[N] = ("Mode", Color, "FontName", FontSize, Angle, interiorAlignment, backgroundColor, rotationPoint);
BottomTics[N] = ("Mode", Color, "FontName", FontSize, Angle, interiorAlignment, backgroundColor, rotationPoint);
LeftTics[N] = ("Mode", Color, "FontName", FontSize, Angle, interiorAlignment, backgroundColor, rotationPoint);
RightTics[N] = ("Mode", Color, "FontName", FontSize, Angle, interiorAlignment, backgroundColor, rotationPoint);
```

The **Tics** parameter specifies the label attributes for the tic marks displayed along a given axis. The tic labels are generated automatically based on the other parameter settings, and are displayed using the given label attributes in the **Tics** parameter. If any attribute is not defined, any previous value of that attribute will be used.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interiorAlignment</td>
<td>Specifies the alignment to use in text strings that contain multiple lines.</td>
</tr>
<tr>
<td>backgroundColor</td>
<td>Background color of the tic label area</td>
</tr>
<tr>
<td>rotationPoint</td>
<td>For rotated axis labels, anchor point for the rotation</td>
</tr>
</tbody>
</table>
The legal values for `interiorAlignment` are LEFT, RIGHT, or CENTER.

The legal values for `rotationPoint` are LEFT, RIGHT.

**Used in These Charts**
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**
```
TopTics = ("ON", black, "Helvetica", 10, LEFT);
```

**Attributes**

<table>
<thead>
<tr>
<th>Mode</th>
<th>Color</th>
<th>FontName</th>
<th>FontSize</th>
<th>Angle</th>
</tr>
</thead>
<tbody>
<tr>
<td>InteriorAlignment</td>
<td>BackgroundColor</td>
<td>RotationPoint</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Mode**

`Mode` determines whether or not the tic labels are shown on that axis.

**Example:**
```
TopTics = ("ON", black, "Helvetica", 10);
```

**Values**

| ON       | Show tic labels for this axis |
| OFF      | Don’t show tic labels on this axis |

**Default**

ON

**TicLayout**

```
TopTicLayout[N] = (Mode, SkipCount, StaggerLevels);
BottomTicLayout[N] = (Mode, SkipCount, StaggerLevels);
LeftTicLayout[N] = (Mode, SkipCount, StaggerLevels);
RightLayout[N] = (Mode, SkipCount, StaggerLevels);
```

The `TicLayout` parameter is normally used in cases where the tic axis labels may overlap if not adjusted. This allows one to adjust the visual data densities for your chart.

**Used in These Charts**
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**
```
TopTicLayout = (AUTOSTAGGER, 5, 3);
```
Mode controls the type of layout mode that should be used with the axis labels.

**Example:**

```
TopTicLayout = (AUTOSKIP, 5, 3);
```

**Values**

- **NORMAL**: No explicit layout processing should occur
- **AUTO**: Insures that labels never overlap. If labels are overlapping, it automatically staggers labels up to the number of levels defined in the `StaggerLevels` parameter (default = 2). If the labels still overlap, even after staggering, labels will be automatically skipped at a constant interval until none overlap.
- **AUTOSKIP**: The axis labels should be automatically skipped at a constant interval if (and only if) they overlap
- **AUTOSTAGGER**: The axis labels should be automatically staggered up to the number of levels defined in the `StaggerLevels` parameter, if (and only if) they overlap
- **SKIP**: a certain number of axis tics should be skipped, and uses the `SkipCount` parameter to determine that number. (Default = 1.)
- **STAGGER**: Axis labels should be staggered, using the number of levels defined in the `StaggerLevels` parameter. (Default = 2.)
- **SKIPSTAGGER**: Axis labels should be skipped and staggered, using the `SkipCount` and `StaggerLevels` parameters. (Defaults = 1 & 2, respectively.)

**Default**

NORMAL

**StaggerLevels**

The `StaggerLevels` parameter defines the number of visual "levels" to which axis labels can or should be staggered for good visibility.

**Example:**

```
TopTicLayout = (AUTOSKIP, 5, 3);
```

**Values**

- **0 or 1**: No stagger occurs
- **>=2**: Integer Number of text lines staggered

**Default**

2
**SkipCount**

SkipCount controls the number of tics that should be skipped should the Mode be set to SKIP.

**Example:**

```plaintext
TopTicLayout = (AUTOSKIP, 5, 3);
```

**Values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No skipping occurs</td>
</tr>
<tr>
<td>&gt;0</td>
<td>Integer Number of axis tics skipped</td>
</tr>
</tbody>
</table>

**Default**

1

**TitleSpacing**

TitleSpacing = Number;

The TitleSpacing parameter is specified in pixels and defines the amount of space between an extended title and the border of the chart. The default value is 5. A value of 0 will allow the title background to extend to the border of the chart. The parameter is only relevant when a title has its extend attribute set ON.

**Used in These Charts**

All

**TopActiveLabels**

TopActiveLabels = ("Label", "URL", "Target"), ...;

The top axis labels become active labels when TopActiveLabels parameter is used. Each set in parenthesis has a corresponding set within a DataSet parameter.

**Used in These Charts**

All

**Attributes**

| Label | URL | Target |

**ToggleDataVisibility**

ToggleDataVisibility = ON | OFF;
When `ToggleDataVisibility = ON`, clicking on a legend item will temporarily hide or show the associated data series. This parameter is only valid in SVG or SVGweb output formats.

**Example:**
```
ToggleDataVisibility = OFF;
```

**Attributes**

(Switch)

### TopAxis

```
TopAxis = (Label, Color, FontName, FontSize, Angle, interiorAlignment);
```

If `TopAxis` is defined for a Combo chart, then the top axis will be used to map the X data values for all line sets, unless otherwise specified using the `LineAxis` parameter. The group sets the typographic characteristics for the data values.

<table>
<thead>
<tr>
<th>interiorAlignment</th>
<th>Specifies the alignment to use in text strings that contain multiple lines.</th>
</tr>
</thead>
</table>

The legal values for `interiorAlignment` are LEFT, RIGHT, or CENTER.

**Used in These Charts**

Combo

**Example:**
```
TopAxis = ("", black, "TimesRoman", 16, 0);
```

**Attributes**

Label    Color    FontName    FontSize    Angle    interiorAlignment

### TopAxisTitle

```
TopAxisTitle = (Label, Color, FontName, FontSize, Angle, interiorAlignment, exteriorAlignment);
```

The `TopAxisTitle` parameter specifies the label attributes for the axis title, which centered along the top axis, just above the grid. When the `Header` parameter, whether because of the use of a legend, or for some other reason, creates a title that seems visually unbalanced, you may find this parameter produces a more pleasing chart title.

<table>
<thead>
<tr>
<th>interiorAlignment</th>
<th>Specifies the alignment to use in text strings that contain multiple lines.</th>
</tr>
</thead>
<tbody>
<tr>
<td>exteriorAlignment</td>
<td>Specifies the alignment for the entire Title object.</td>
</tr>
</tbody>
</table>

The legal values for `interiorAlignment` and `exteriorAlignment` are LEFT, RIGHT, or CENTER.
**Used in These Charts**
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**
```
TopAxisTitle = ("Ceres Prototype Project Schedule
", black, "Helvetica", 12);
```

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>Color</th>
<th>FontName</th>
<th>FontSize</th>
<th>Angle</th>
<th>interiorAlignment</th>
<th>exteriorAlignment</th>
</tr>
</thead>
</table>

**TopAxisTitleActiveLabel**

```
TopAxisTitleActiveLabel = ("Label", "URL", "Target");
```

TopAxisTitleActiveLabel defines a single active label destination for the TopAxisTitle parameter.

**Used in These Charts**
All

**Example:**
```
TopAxisTitleActiveLabel = ("Destination", "demo.html", "frame1");
```

**Attributes**

<table>
<thead>
<tr>
<th>Label</th>
<th>URL</th>
<th>Target</th>
</tr>
</thead>
</table>

**TopAxisTitleBox**

```
TopAxisTitleBox = (Color, BorderType, BorderWidth, "ImageURL", ImageFormat, BorderColor, TRCornerStyle, BRCornerStyle, BLCornerStyle, CornerColor); 
```

The TopAxisTitleBox parameter specifies the region attributes for the axis title centered along the axis. The image-related attributes need not be used unless you want to use an image texture on the box.

**Used in These Charts**
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**
```
TopAxisTitleBox = (lightgray, SHADOW, 3,,gray);
```

**Attributes**

| Color | BorderType | BorderWidth | ImageURL | ImageFormat | BorderColor | TRCornerStyle | BRCornerStyle | BLCornerStyle | CornerColor |
|-------|------------|-------------|----------|-------------|-------------|----------------|---------------|---------------|-------------|-------------|
**XXCornerStyle**

The drawing style for each of the four corners of the region. Styles are specified in a clockwise fashion starting in the upper left - Top Left, Top Right, Bottom Right, and Bottom Left. Legal values are SQUARE, SNIP and ROUND. The default is SQUARE.

**TopColor**

TopColor = Color;

*TopColor* controls the color of the top axis and the tic marks, but not the tic mark labels. The default axis color is black. If the NULL color is specified, the axis color is not changed by this parameter.

**Used in These Charts**

Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**

TopAxisColor = xB5D5F0;

**Attributes**

Color

**TopDrawMinorTics**

TopDrawMinorTics = ON | OFF;

*TopDrawMinorTics* controls whether or not Top tics are drawn. The default value is ON.

**Example:**

TopDrawMinorTics = OFF;

**Attributes**

(Switch)

**TopFormat**

TopFormat = (FormatType, "FormatExpr", "TimeBase", "TimeUnit");

*TopFormat* adjusts the numeric labels that are automatically generated for the top axis, should one be defined.
### Used in These Charts
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

### Example:
```
TopFormat = (DATE, "%w\n%M/%D", "1 Jan 2000", "1h");
TopFormat = (DATE, "%M/%D", "1 Apr 96", "1d");
TopFormat = (INTEGER);
TopFormat = (FLOAT,"$%,9.2f",,);
```

### Attributes

<table>
<thead>
<tr>
<th>FormatType</th>
<th>FormatExpr</th>
<th>TimeBase</th>
<th>TimeUnit</th>
</tr>
</thead>
</table>

FormatType specifies the type of number being displayed on the top axis.

### Example:
```
TopFormat = (DATE, "%w\n%M/%D", "1 Jan 2000", "1h");
TopFormat = (DATE, "%M/%D", "1 Apr 96", "1d");
TopFormat = (INTEGER);
TopFormat = (FLOAT,"$%,9.2f",,);
```

### Values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATE</td>
<td>Axis value are shown as date and/or time values. See Appendix A: Date and Time Values for further detail.</td>
</tr>
<tr>
<td>FLOAT</td>
<td>Axis values are shown with decimal parts.</td>
</tr>
<tr>
<td>INTEGER</td>
<td>Axis values are shown as integers, and are rounded if necessary.</td>
</tr>
</tbody>
</table>

### Default

INTEGER

**TimeBase**

The TimeBase attribute specifies the base date to be used when determining the actual date or time value when using a time unit or numeric value. See Appendix A: Date and Time Values for further detail.

### Example:
```
TopFormat = (DATE, "%w\n%M/%D", "1 Jan 2000", "1h");
TopFormat = (DATE, "%M/%D", "1 Apr 96", "1d");
```

### Values

String values representing dates or times

### Default

None
TimeUnit

The `TimeUnit` attribute controls the time multiplier to be used when determining the actual data/time value when using a numeric value. See Appendix A: Date and Time Values for further detail.

Example:

```
TopFormat = (DATE, "%w\n%M/%D", "1 Jan 2000", "1h");
TopFormat = (DATE, "%M/%D", "1 Apr 96", "1d");
```

Values

String values representing dates or times

Default

None

TopLabels

```
TopLabels = "Labell", "Label2", ...;
```

The `TopLabels` parameter specifies a list of custom tic mark labels that will be used instead of the numeric labels automatically generated by the axis. The `TopLabels` will be evenly placed along the axis, overriding any tic placement specified by the `StepValue` attribute.

Used in These Charts

Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

Example:

```
            "Paul";
```

Attributes

Label

TopMargins

```
TopMargins = (LeftSideMargin, RightSideMargin);
```

The `TopMargins` parameter specifies the gap, in pixels, at the beginning and end of the top axis. Most often used to prevent clipping of data points at the extreme ends of the scale.

Example:

```
TopMargins = (20, 20);
```
TopScale

TopScale = (MinValue, MaxValue, StepValue);

The TopScale parameter specifies the minimum and maximum data values which will be displayed along the top axis. If the TopScale parameter is not defined, or the MinValue and MaxValue parameters are the same or one of them is not defined, then the tic mark locations will be automatically determined based on the actual data values being displayed. That is, the axis will be "autoscaled" using the current data values to determine "reasonable" values for MinValue, MaxValue and StepValue. If values are supplied for any of MinValue, MaxValue, or StepSize, those values will be used as part of the autoscaling.

Used in These Charts
Bar, Box, Combo, Bubble, Line, Pareto, Stock, Time, X-Y

Example:
TopScale = TopScale = ("1 Apr 96", "1 Jun 96", "14d");

Attributes
MinValue MaxValue StepValue

MinValue

MinValue sets the absolute lower visible limit for the top axis scale.

Example:
TopScale = ("1 Apr 96", "1 Jun 96", "14d");

Values
Any real number, date, or time less than MaxValue

Default
None

MaxValue

ScrollMax sets the absolute upper visible limit for the top axis scale.

Example:
TopScale = ("1 Apr 96", "1 Jun 96", "14d");

Values
Any real number, date, or time greater than MinValue
**Default**
None

**StepValue**

StepValue is optional, and may be used to specify a given step between tic marks along the top axis, starting with the MinValue. If StepValue is not an even multiple of the difference between the MinValue and MaxValue, then no tic mark will be displayed at the MaxValue. If StepValue is not defined, tic marks will be placed at "reasonable" locations along the axis, depending on the range of values being displayed.

**Example:**
```plaintext
TopScale = ("1 Apr 96", "1 Jun 96", "14d");
```

**Values**
Any real number, date, or time between MinValue and MaxValue

**Default**
1

**TopScroll**

TopScroll = (ScrollMin, ScrollMax);

The TopScroll parameter specifies a range of values through which an axis can be scrolled. When the ScrollMin and ScrollMax attributes are defined for the axis, the axis will be displayed as a slider bar, using the axis color defined, with a white background. The relative size of the slider represents the percentage of the entire range currently being displayed. That is, it graphically depicts the size of the current axis range (MinValue and MaxValue attributes) relative to the scrollable region (ScrollMin and ScrollMax attributes). See the TopScale parameter for MinValue and MaxValue definitions.

TopScroll should only be used in conjunction with the TopScale parameter.

**Example:**
```plaintext
TopScroll = (0, 98);
```

**Attributes**

ScrollMin
ScrollMax

**ScrollMin**

ScrollMin sets the lower visible limit for a scrollbar defined with TopScroll

**Example:**
```plaintext
TopScroll = (0, 98);
```

**Values**

<MinValue
**Common CDL Attributes**

**ScrollMax**

ScrollMax sets the upper visible limit for a scrollbar defined with TopScroll.

*Example:*

```
TopScroll = (0, 98);
```

**Values**

>MaxValue

*Default*

None

**TopTics**

```
TopTics = ("Mode", Color, "FontName", FontSize, Angle, interiorAlignment, backgroundColor, rotationPoint);
```

The TopTics parameter specifies the label attributes for the tic marks displayed along the axis. The tic labels are generated automatically based on the other parameter settings, and are displayed using the given label attributes in the TopTics parameter. If any attribute is not defined, any previous value of that attribute will be used.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interiorAlignment</td>
<td>Specifies the alignment to use in text strings that contain multiple lines.</td>
</tr>
<tr>
<td>backgroundColor</td>
<td>Background color of the tic label area</td>
</tr>
<tr>
<td>rotationPoint</td>
<td>For rotated axis labels, anchor point for the rotation</td>
</tr>
</tbody>
</table>

The legal values for `interiorAlignment` are LEFT, RIGHT, or CENTER.

The legal values for `rotationPoint` are LEFT or RIGHT.

**Used in These Charts**

Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

*Example:*

```
TopTics = ("ON", black, "Helvetica", 10);
```

**Attributes**

<table>
<thead>
<tr>
<th>Mode</th>
<th>Color</th>
<th>FontName</th>
<th>FontSize</th>
<th>Angle</th>
<th>interiorAlignment</th>
<th>InteriorAlignment</th>
<th>backgroundColor</th>
<th>RotationPoint</th>
</tr>
</thead>
</table>

© Visual Mining, Inc 2002-2011

www.visualmining.com
Common CDL Attributes

Mode

Mode determines whether or not the tic labels are shown on that axis.

Example:

```
TopTics = ("ON", black, "Helvetica", 10);
```

Values

<table>
<thead>
<tr>
<th>ON</th>
<th>Show tic labels for this axis</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFF</td>
<td>Don’t show tic labels on this axis</td>
</tr>
</tbody>
</table>

Default

ON

TopTicLayout

```
TopTicLayout = (Mode, SkipCount, StaggerLevels);
```

The TopTicLayout parameter is normally used in cases where the tic axis labels may overlap if not adjusted. This allows one to adjust the visual data densities for your chart.

Used in These Charts

Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

Example:

```
TopTicLayout = (AUTOSTAGGER, 5, 3);
```

Attributes

<table>
<thead>
<tr>
<th>Mode</th>
<th>SkipCount</th>
<th>StaggerLevels</th>
</tr>
</thead>
</table>

Mode

Mode controls the type of layout mode that should be used with the axis labels.

Example:

```
TopTicLayout = (AUTOSKIP, 5, 3);
```
Common CDL Attributes

Values

NORMAL  No explicit layout processing should occur
AUTO   Insures that labels never overlap. If labels are overlapping, it automatically staggers labels up to
       the number of levels defined in the StaggerLevels parameter (default = 2). If the labels
       still overlap, even after staggering, labels will be automatically skipped at a constant
       interval until none overlap.
AUTOSKIP The axis labels should be automatically skipped at a constant interval if (and only if) they
overlap
AUTOSTAGGER The axis labels should be automatically staggered up to the number of levels defined in the
     StaggerLevels parameter, if (and only if) they overlap
SKIP    a certain number of axis tics should be skipped, and uses the SkipCount parameter to determine
     that number. (Default = 1.)
STAGGER Axis labels should be staggered, using the number of levels defined in the StaggerLevels
     parameter. (Default = 2.)
SKIPSTAGGER Axis labels should be skipped and staggered, using the SkipCount and StaggerLevels
     parameters. (Defaults = 1 & 2, respectively.)

Default
NORMAL

StaggerLevels

StaggerLevels the number of visual "levels" to which axis labels can or should be staggered for good
visibility.

Example:

TopTicLayout = (AUTOSKIP, 5, 3);

Values

0 or 1 No stagger occurs
>=2   Integer Number of text lines staggered

Default
2

SkipCount

SkipCount controls the number of tics that should be skipped should the Mode be set to SKIP.

Example:

TopTicLayout = (AUTOSKIP, 5, 3);

Values

0  No skipping occurs
>0  Integer Number of axis tics skipped
**Default**

1

**TopTicLength**

TopTicLength = Number;

The TopTicLength parameter defines the size of axis tic marks which are displayed along the top axis of a chart. The parameter will reset the automatically generated tic length. The value defines the number of pixels to use for the length of the tic mark. By default, the number of pixels used is the width of the character zero (0) as found in the font applied to the label. Setting the TopTicLength to the value -1 will cause the default size to be used.

**Attributes**

Number

**Number**

Apparent length of a top axis tic mark in a chart, in pixels.

**Used in These Charts**

Bar, Box, Combo, Bubble, Line, Pareto, Stock, Strip, Time, X-Y

**Example:**

TopTicLength = 10;

**Values**

0

No effect (zero length tics are not drawn).

1 or greater

Whole number length in pixels

**Default**

-1

**Update**

Update;

Update causes one "slot" of data to be extracted from the input data queue for each data set and displayed. Update will also update the axes and configure the dwell labels for the slot (if defined).

**Used in These Charts**

Strip
Example:

```
Update;
```

**Attributes**

None

---

**UniqueTaskColors**

```
UniqueTaskColors = ON|OFF;
```

If set to ON, then a unique color will be chosen from the color table for each task in a taskbar. (See the ColorTable parameter.) The default is OFF which uses the previously specified taskbar color.

**Used in These Charts**

*Time*

Example:

```
UniqueTaskColors = ON;
```

**Attributes**

mode

---

**WhiskerType**

```
WhiskerType = Type;
```

WhiskerType controls the width of the whisker on a box. The whisker can be drawn as either a line or a box.

**Used in These Charts**

*Box Chart*

Example:

```
WhiskerType = BOX;
WhiskerType = LINE;
```

**Attributes**

Type
**Type**

Type refers to the width of the whisker on a box.

**Values**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOX</td>
<td>BOX is 40% of the main box width</td>
</tr>
<tr>
<td>Line</td>
<td>Line whiskers are 1 pixel</td>
</tr>
</tbody>
</table>

**Default**

BOX

**TwentyLineSetName**

TwentyLineSetName = *name*;

Name assigned to the 20% line. Used in the legend if the Legend CDL parameter does not define a label for the 20% line.

**Used in These Charts**

Pareto

**Example:**

TwentyLineSetName = “20% Line”

**ViewPoint**

ViewPoint = (CARTESIAN, X, Y, Z);

or

ViewPoint = (SPHERICAL, radius, phi, theta);

The viewpoint from which a 3d barchart is rendered. The viewpoint can only be in front, on top, and to the right of the center of the chart.

**Used in These Charts**

3DBarchart

**Example:**

ViewPoint = (CARTESIAN, 300, 0, 300);

ViewPoint = (SPHERICAL, 0, 45, 300);

**Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
<th>Radius</th>
<th>Phi</th>
<th>Theta</th>
</tr>
</thead>
</table>

**CoordinateSystem**

X, Y, Z, Radius, Phi, Theta
**CoordinateSystem**

The system in which the specified numeric values are defined. Legal values are CARTESIAN and SPHERICAL.

If the CoordinateSystem is CARTESIAN, then the numeric values that follow are the X, Y and Z coordinates of the viewpoint for the chart. These coordinates are specified relative to a center point of the chart.

If the CoordinateSystem is SPHERICAL, then the numeric values that follow are the radius, angle of elevation (phi) and the angle of rotation (theta).

**X,Y,Z**

The location of a 3dbarchart viewpoint when the coordinate system is CARTESIAN. These numbers are relative to size of the chart. Only positive values are valid in Cartesian coordinates. Setting any of the coordinates to a negative number is the same as setting that coordinate to 0. To look at the front of the chart, set the view point to (0, 0, a). To look at the chart from a 45 degree angle, set the view point to (a, 0, a). The greater a is, the further the camera is from the chart.

**Radius**

The distance the viewpoint is from the center of chart when the coordinate system is SPHERICAL. Radius should be specified as a positive integer. The vmagnitude of the radius is relative to the width and height of the chart.

**Phi**

The angle of elevation of the viewpoint when coordinate system is SPHERICAL. Phi should be specified as an angle between 0 and 90. 0 is the angle looking directly at the front of the chart. 90 is the angle looking directly down at the top of the chart.

**Theta**

The angle of rotation of the viewpoint when coordinate system is SPHERICAL. Theta should be specified as an angle between 0 and 90. 0 is the angle looking directly at the front of the chart. 90 is the angle looking directly at the end of the chart.

**ZAxisLabels**

ZAxisLabels = (ON|OFF, Color, FontName, FontSize, Angle, interiorAlignment)

The drawing style for Z axis labels in a 3D Barchart.
Used in These Charts
3DBarchart

Example:

\[ Z\text{AxisLabels} = ("ON","black","Verdana Plain",12,0,null); \]

**AxisZoom**

```
BottomZoom[N] = ON|OFF;
LeftZoom[N] = ON|OFF;
RightZoom[N] = ON|OFF;
TopZoom[N] = ON|OFF;
```

*AxisZoom* permits one to zoom into a portion of a chart. When *AxisZoom = ON*, clicking and dragging a rectangle over a chart will zoom with respect to that axis on the chart. This zoom operation does not require the axis to be scrollable.

Example:

```
LeftZoom = ON;
```

**Attributes**

*(Switch)*

**RubberbandBorderStyle**

```
RubberbandBorderStyle = (linetype, linewidth, linecolor);
```

*RubberbandBorderStyle* defines the borderstyle to apply to the rubberbanded box created by clicking and dragging to create zoom rectangle.

Used in These Charts
All

Example:

```
RubberbandBorderStyle = (SOLID,1,YELLOW);
```

**Attributes**

*LineType*  *LineWidth*  *LineColor*
Common CDL Attributes

Values

<table>
<thead>
<tr>
<th>Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOLID</td>
<td>A solid line is displayed (default).</td>
</tr>
<tr>
<td>DOTTED</td>
<td>A dotted line is displayed.</td>
</tr>
<tr>
<td>DASHED</td>
<td>A dashed line is displayed.</td>
</tr>
<tr>
<td>DOTDASH</td>
<td>Alternating dots and dashes are displayed.</td>
</tr>
</tbody>
</table>

Default

SOLID

LineWidth

The width of the line defaults to 1 pixel.

LineColor

The LineColor defaults to black.

RubberbandFill

RubberbandFill = Color;

Defines the color of the rubberbanded box created by clicking and dragging a zoom rectangle.

Used in These Charts

All

Example:

```plaintext
RubberBandFill = YELLOW_100);
```

Attributes

Color

Default

NONE

8. Common CDL Attributes

Many of the CDL attributes are shared across CDL parameters. Rather than repeat descriptions of these attributes ad nauseam in the reference listings, we are listing their details here only once. Throughout the
rest of the guide, the attribute list for each CDL element points to these common attributes when the attribute name is in *italic*. When you see an attribute listed in *italic*, it means that you should look to this section for specific details about the attribute. If you recognize a term from the list of attributes-in-common, but it is not shown in italic, that means that there is some element-specific information about the attribute, so the description is provided with the element.

In the Examples, the attribute is shown in **bold**, to help you locate it. In actual code, attributes would not be bolded.

**Angle**

*Angle* defines the angle of counterclockwise rotation, in degrees, for an entire label.

**Example:**

```
LeftTitle = ("Rotated Label", black, "TimesRoman", 16, 90, LEFT);
```

**Values**

- 0: degrees counterclockwise from horizontal
- 90: degrees counterclockwise from horizontal
- 180: degrees counterclockwise from horizontal
- 270: degrees counterclockwise from horizontal

**Default**

0

**BorderColor**

*BorderColor* defines the color used to draw a region border when the BOX value of *BorderType* is used.

**Example:**

```
RightTitleBox = (white, BOX, 2, green);
```

**Values**

Any valid *Color* value may be used. See below for *Color* definitions.

**Default**

black

**BorderType**

*BorderType* defines the edges of a visual region such as a background or label. When the BOX and SHADOW styles are specified, the border color defaults to black. When the RAISED and RECESS styles are specified, the border color will be chosen based on the region color.
Common CDL Attributes

BorderWidth

Example:

\[
\text{LeftTitleBox} = (\text{yellow, RAISED, 10});
\]

Values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>No border</td>
</tr>
<tr>
<td>BOX</td>
<td>Simple Box outline</td>
</tr>
<tr>
<td>SHADOW</td>
<td>Shadow border</td>
</tr>
<tr>
<td>RAISED</td>
<td>Raised border</td>
</tr>
<tr>
<td>RECESS</td>
<td>Recessed border</td>
</tr>
</tbody>
</table>

Default

NONE

BorderWidth

BorderWidth defines the width of the edges of a visual region such as a background or label.

Example:

\[
\text{LeftTitleBox} = (\text{yellow, RAISED, 10});
\]

Values

Integer number of pixels wide

Color

Colors can be specified in any NetCharts parameter using any of the following values:

- Predefined Color Name
- Hexadecimal RGB value
- NULL Keyword

These colors may be rendered differently on different platforms or within different browsers because of differences in system or browser defined color tables.

Example:

\[
\text{BarBorder} = (\text{SOLID, 2, red});
\]

Values

Predefined color names are the most convenient method for specifying a color. Over 100 names are supported. Table 1, below is a list of the predefined color names. The default color palette in NetCharts Designer show these predefined colors. You may also use these:
**Common CDL Attributes**

xRGGGBB  \( \text{RR, GG, BB} \) are the hexadecimal values for red, green, and blue, in the range 00 to ff. The leading ‘x’ is mandatory and designates the beginning of the hexadecimal value.

**NULL**

May also be used depending on the specific chart parameter being defined. For example, the `DataSets` parameter in the `NFBarchartApp` accepts a NULL parameter, whereas the `Background` parameter does not accept NULL colors.

Translucency can be added to any color specification by appending a translucency qualifier between 0 and 250. For example `red_0` or `xFF0000_0` is completely translucent, while `red_250` and `xFF0000_250` both represent solid red.

**Default**

Black for text, gray for backgrounds and chart elements

**Table 1: Predefined Color Names Used in Visual Mining Applets**

<table>
<thead>
<tr>
<th>Name</th>
<th>Name</th>
<th>Name</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>antiquewhite</td>
<td>dimgray</td>
<td>magenta</td>
<td>salmon</td>
</tr>
<tr>
<td>aquamarine</td>
<td>dodgerblue</td>
<td>maroon</td>
<td>slategray</td>
</tr>
<tr>
<td>azure</td>
<td>firebrick</td>
<td>mediumaquamarine</td>
<td>sandybrown</td>
</tr>
<tr>
<td>beige</td>
<td>floralwhite</td>
<td>mediumblue</td>
<td>saddlebrown</td>
</tr>
<tr>
<td>bisque</td>
<td>forestgreen</td>
<td>mediumpurple</td>
<td>seagreen</td>
</tr>
<tr>
<td>black</td>
<td>fuchsia</td>
<td>mediumseagreen</td>
<td>seashell</td>
</tr>
<tr>
<td>blue</td>
<td>gainsboro</td>
<td>mediumvioletred</td>
<td>silver</td>
</tr>
<tr>
<td>blueviolet</td>
<td>ghostwhite</td>
<td>mediumvioletgreen</td>
<td>skyblue</td>
</tr>
<tr>
<td>brown</td>
<td>goldenrod</td>
<td>mediumspringgreen</td>
<td>slateblue</td>
</tr>
<tr>
<td>burlywood</td>
<td>gold</td>
<td>mediumturquoise</td>
<td>slateblue</td>
</tr>
<tr>
<td>cadetblue</td>
<td>gray</td>
<td>midnightblue</td>
<td>snow</td>
</tr>
<tr>
<td>chartreuse</td>
<td>green</td>
<td>mintcream</td>
<td>springgreen</td>
</tr>
<tr>
<td>chocolate</td>
<td>greenyellow</td>
<td>moccasin</td>
<td>steelblue</td>
</tr>
<tr>
<td>coral</td>
<td>honeydew</td>
<td>navy</td>
<td>tan</td>
</tr>
<tr>
<td>cornflowerblue</td>
<td>hotpink</td>
<td>oldlace</td>
<td>tomato</td>
</tr>
<tr>
<td>cornsilk</td>
<td>indigo</td>
<td>olive</td>
<td>teal</td>
</tr>
<tr>
<td>crimson</td>
<td>indianred</td>
<td>olivedrab</td>
<td>turquoise</td>
</tr>
<tr>
<td>cyan</td>
<td>ivory</td>
<td>orange</td>
<td>violet</td>
</tr>
<tr>
<td>darkblue</td>
<td>khaki</td>
<td>orangered</td>
<td>wheat</td>
</tr>
<tr>
<td>darkcyan</td>
<td>lavenderblush</td>
<td>orchid</td>
<td>whitesmoke</td>
</tr>
<tr>
<td>darkgoldenrod</td>
<td>lawngreen</td>
<td>palegoldenrod</td>
<td>white</td>
</tr>
<tr>
<td>darkgray</td>
<td>lemonchiffon</td>
<td>palevioletred</td>
<td>yellow</td>
</tr>
<tr>
<td>darkgreen</td>
<td>lightblue</td>
<td>palegreen</td>
<td>yellowgreen</td>
</tr>
<tr>
<td>darkkhaki</td>
<td>lightcoral</td>
<td>paleturquoise</td>
<td></td>
</tr>
<tr>
<td>darkmagenta</td>
<td>lightgoldenrodyellow</td>
<td>papayawhip</td>
<td></td>
</tr>
<tr>
<td>darkorchid</td>
<td>lightgrey</td>
<td>peachpuff</td>
<td></td>
</tr>
<tr>
<td>darkorange</td>
<td>lightgreen</td>
<td>peru</td>
<td></td>
</tr>
<tr>
<td>darkred</td>
<td>lightpink</td>
<td>pink</td>
<td></td>
</tr>
<tr>
<td>darksalmon</td>
<td>lightsalmon</td>
<td>plum</td>
<td></td>
</tr>
<tr>
<td>darkseagreen</td>
<td>lightseagreen</td>
<td>powderblue</td>
<td></td>
</tr>
<tr>
<td>darkslateblue</td>
<td>lightskyblue</td>
<td>purple</td>
<td></td>
</tr>
<tr>
<td>darkslategray</td>
<td>lightslategray</td>
<td>red</td>
<td></td>
</tr>
<tr>
<td>darkturquoise</td>
<td>lightsteelblue</td>
<td>rosybrown</td>
<td></td>
</tr>
<tr>
<td>darkviolet</td>
<td>lightyellow</td>
<td>royalblue</td>
<td></td>
</tr>
<tr>
<td>darkolivegreen</td>
<td>lime</td>
<td></td>
<td></td>
</tr>
<tr>
<td>deppink</td>
<td>limegreen</td>
<td></td>
<td></td>
</tr>
<tr>
<td>deepskyblue</td>
<td>linen</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**CDL Reference Manual – 7.1**

© Visual Mining, Inc 2002-2011
www.visualmining.com
**FontName**

*FontName* describes the font style to be used for the label. The exact list of font styles is platform dependent, but the font styles below are guaranteed to be available for any Java implementation.

Font names can be augmented with additional font style information. Adding "Plain", "Bold" or "Italic" to the font name modifies the style as specified. For example "Courier Bold Italic" is a valid font name specification. By default NetCharts chooses a BOLD style for the specified font so "TimesRoman" is equivalent to "TimesRoman Bold". To get a standard version of a font, add "Plain" to the name.

Font names can also be augmented with "underline", "overline", "linethrough", "ascent=N", "descent=N", "leading=N" and "maxLineAdvance=N".

Any combination of style modifiers is allowed. For example "TimesRoman Plain Italic underline overline linethrough ascent=10 descent=0 leading=-15" is a valid font specification which uses a nonbold italic type, draws a line above, below and through the text, and controls the amount of space allocated for character ascents, descents and the space between consecutive lines.

*maxLineAdvance* specifies the maximum length (in pixels) NetCharts will allow a string using this font specification to be before it wraps to a new line.

**Example:**

```java
RightTitle = ("Large\nLabel", blue, "Courier Plain underline leading=5", 12);
```

**Values**

<table>
<thead>
<tr>
<th>Java font</th>
<th>X-11 font</th>
<th>Windows Font</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>misc-fixed</td>
<td>Arial</td>
</tr>
<tr>
<td>Helvetica</td>
<td>adobe-helvetica</td>
<td>Arial</td>
</tr>
<tr>
<td>TimesRoman</td>
<td>adobe-times</td>
<td>Times New Roman</td>
</tr>
<tr>
<td>Courier</td>
<td>adobe-courier</td>
<td>Courier New</td>
</tr>
<tr>
<td>Dialog</td>
<td>b&amp;h-lucida</td>
<td>MS Sans Serif</td>
</tr>
</tbody>
</table>

**Default**

TimesRoman

**FontSize**

*FontSize* defines the point size of the font.

**Example:**

```java
RightTitle = ("Large\nLabel", blue, "Courier", 12);
```

**Values**

The available values depend on the font style defined and the platform.

**Default**

12

20 pt font size
FormatExpr

The FormatExpr attribute specifies a numeric display format to be used for axis or pie slice labels and the active labels generated by default. The format allows a developer to specify numeric and date/time formats using expressions similar to those provided in the C/Unix printf function. The format expressions consist of an arbitrary string, plus one or more occurrences of format components, which are preceded by ‘%’.

Example:

```
LeftFormat = (FLOAT, "$%,9.2f", ,);
SliceFormat = (FLOAT, "%5.2f");
```

<table>
<thead>
<tr>
<th>FORMAT</th>
<th>RESULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>%g</td>
<td>1234.456</td>
</tr>
<tr>
<td>%f</td>
<td>1234.46</td>
</tr>
<tr>
<td>%,f</td>
<td>1,234.46</td>
</tr>
<tr>
<td>%d</td>
<td>1234</td>
</tr>
<tr>
<td>%,d</td>
<td>1,234</td>
</tr>
<tr>
<td>%8.1f</td>
<td>1234.5</td>
</tr>
<tr>
<td>%08.1f</td>
<td>001234.5</td>
</tr>
<tr>
<td>Cost=$%,.1fM</td>
<td>Cost=$1,234.5M</td>
</tr>
</tbody>
</table>

Values

| %f | Display numeric value with fixed number of decimal digits, which can be zero. |
| %g | Display numeric value using platform specific default format. Generally, this results in a "reasonable" format, unless a large number of decimal digits exist or the number is very large or very small. |
| %d | Display numeric value as an integer, rounding if necessary. |

The following modifiers can be applied to the format component to yield different outputs:

```
-,0WW.DD
```

where all parts are optional and are interpreted as follows:

- Left justify the result, otherwise right justify it.
- Separate the whole number into groups using the group symbol. By default, a comma is used as the group symbol and the group size is set to 3. This option is ignored if zero padding is also specified.
0 Pad with leading zeros, if necessary, to fill entire field.
WW Total width of the formatted field. If the field is naturally wider than this value, it is ignored.
. Separate the whole number from the decimal number using the decimal symbol. By default, a period is used as the decimal symbol.
DD Number of decimal digits to display (for %f format only). If not specified, the default number is 2.

Default

%.2f
**ImageFormat**

ImageFormat specifies how the image will be displayed in the region.

**Example:**

```
HeaderBox = (lightblue, RAISED, 5, "$IMAGES/nc220x90.gif", CENTER);
```

**Values**

- **TILE**  
  Tiles or clips the image if not the same size as the region.
- **SIZE**  
  Scales the image to the size of the region.
- **CENTER**  
  Image is centered in the region

**Default**

TILE

**ImageURL**

This attribute can be used to specify an image file to be used to fill the region. Any valid URL may be specified. If a relative URL is given, it will be interpreted based on the DocumentBase of the HTML document. See Known Problems for details concerning the limits on the use of images within specific browsers, in some situations.

Depending on your browser environment, files in the $DOCBASE directory hierarchy may not be accessible because of security restrictions. For that reason, data and image files used by all NetCharts examples are located within the $CODEBASE directory hierarchy.

**Example:**

```
RightTitleBox = (lightgreen, SHADOW, 4, "$SYMBOLS/paste.gif", CENTER);
```

**Values**

Relative addressing is supported, or you may use the following keywords for the sake of convenience:

- **$DOCBASE**  
  document base
- **$CODEBASE**  
  code base
- **$NETCHARTS**  
  $CODEBASE/netcharts
- **$ICONS**  
  $NETCHARTS/icons
- **$IMAGES**  
  $NETCHARTS/images
- **$SYMBOLS**  
  $NETCHARTS/symbols
- **$PATTERNS**  
  $NETCHARTS/patterns

These variables are only interpreted if they are used as the first value in a GIF URL. For example, the following URL will cause the chart to look in the "images" sub-directory in the $NETCHARTS classpath for the given file:

```
"$IMAGES/cut.gif"
```

**Default**

None
**Label**

Any text string can be specified for a `Label`, and may include newline (\n) characters for multi-line labels. If the `Label` attribute is not defined or is defined to be "" (the empty string), then a default label will be generated for that data value, depending on the specific chart type. Usually, the default label displays one or more of the numeric values associated with the data point.

In the case of dwell labels, if the first attribute is set to `OUTLINE` then instead of displaying a dwell label, an outline will be drawn around the associated item when the mouse hovers over that item.

**Example:**

```plaintext```
FooterActiveLabel = ("Days", "cfl.html", "frame1");
FooterActiveLabel = ("OUTLINE", "cfoot.html", "body");
```

**Values**

- **Any text string** may include \n
- **""** default label, usually the numeral value of the data
- **ON** display a popup label over the target
- **OUTLINE** display an outline of the target
- **OFF** don't display anything

**Default**

None

**LineType**

`LineType` tells how to draw lines within a parameter belonging to a chart that features lines, such as a combo chart, line chart, stock chart, or strip chart.

**Example:**

```plaintext```
LineStyle     = (DASHED, 3, red), (SOLID, 2, blue);
```

**Values**

- **SOLID** solid lines (default), such as this: ______
- **DOTTED** dotted lines, such as this: ......
- **DASHED** dashed lines, such as this: ------
- **DOTDASH** combined dot and dash lines, something like this: ____. __

**Default**

SOLID
LineWidth

The **LineWidth** attribute allows the specification of the line thickness in pixels, with 1 as the default value. **LineWidth** is specified with a GridLine parameter, and refers to a Grid parameter.

**Example:**
```
GridLine = (VERTICAL, BAR), (HORIZONTAL, DOTTED, 2);
```

**Values**

Integer number of pixels of line width

**Default**

1

Name

The **Name** attribute is a string, in quotes, identifying the particular chart element or data set. **Name** may not actually be seen in a chart except when a legend is used, in which instance it is shown associated with a symbol.

**Example:**
```
AddDataPoint = ("R1", 320, 199, 3.85);
<PARAM NAME=Charts VALUE='"
(Piechart1", PIECHART),
("Piechart2", PIECHART),
("Piechart3", PIECHART);'>
NoteSets = ("note1") , ("note2");
NoteSets = ("Axes", BOTTOM), ("TextOnly", CENTER), ("Text2Only", TOP);
LineSets = ("Sprocket", black),
("Actuator", black),
("Do-Hicky", black),
("Thingy", black);
```

**Values**

Any legal string value

**Default**

None

Target

**Target** is used in the same way that it is used in HTML. If the destination document is to be loaded into a window or frame other than the current window or frame, you can specify where the destination document should load by assigning a window or frame name to the **Target** attribute. Target frame names must be assigned to frames and windows as identifiers.
If Target is not defined, the current frame or window will be updated.

If the value, LOADPARAMS, is used as the target, then the URL will be assumed to point to a NetCharts parameter file, which contains only NetCharts parameter definitions. In that case, the NetCharts parameter file will be processed and all of parameters will be used to update the current chart. This allows users to modify the chart slightly or completely change all of the data, without creating a new chart.

Example:

```
BarActiveLabels = ("Hardware Resources",
http://www.visualmining.com/products/netcharts.html,
_blank);
```

**Values**

Standard HTTP target designation. Case-sensitive identifier when the frame or window name has been assigned via the target element’s NAME attribute. Can use the four reserved HTML target names:

- _blank: Browser creates a new window for the destination document.
- _parent: Destination document replaces the current frame’s framesetting document, if one exists; otherwise, defaults to _self.
- _self: Destination document replaces the current document in its window or frame.
- _top: Destination document is to occupy the entire browser window, replacing any and all framesets that may be loaded.

LOADPARAMS: Points to a NetCharts parameter file, which is specified by the URL attribute.

**Default**

The current frame or window will be updated.

**URL**

URLs are used to indicate the location of a file on the Web or your Intranet.

Example:

```
BarActiveLabels = ("Hardware Resources",
http://www.visualmining.com/products/netcharts.html);
```

```
ActiveLabels1 = ("", "Barchart9Mon.html", "InfoFrame"),
("", "Barchart9Tue.html"),
("", "Barchart9Wed.html"),
("OUTLINE", "Barchart9Thu.html"),
("OUTLINE", "Barchart9Fri.html");
```

**Values**

Standard HTTP URL designation, such as you would use to specify a web page file. If you use full addressing, the normal thing happens. If you use relative addressing, the address is relative to the document containing the applet, not the Codebase of the applet.

**Default**

None
Width

Specifies width of some chart element, in pixels.

Example:
   SliceBorder = (SOLID, 2, red);

Values
Values are in whole numbers of pixels, unless otherwise specified.

Default
None

XAxis

Specifies which side of the chart becomes the X axis.

Example:
   GridAxis = (BOTTOM, LEFT), (TOP, RIGHT);

Values
   TOP          top of chart
   BOTTOM       bottom of chart

Default
   BOTTOM

YAxis

Specifies which side of the chart becomes the Y axis.

Example:
   GridAxis = (BOTTOM, LEFT), (TOP, RIGHT);

Values
   LEFT         left of chart
   RIGHT        right of chart

Default
   LEFT
Appendix A: Date and Time Values

All NetCharts applications provide support for Date/Time values by allowing chart developers to input Date/Time values as data elements, and to configure axes ticks and rollover labels with a variety of formatting options.

Date/Time Data Input

Mapping Date/Time Information

Since Date/Time information is non-numeric, each Date/Time input datum must be transformed into a numeric value in order to properly process the information. The means by which this conversion occurs is based on the type of input date information (absolute date value vs. relative or numeric), whether or not the axis scale is defined or autoscaled, and whether or not a TimeBase has been defined. The following are some general guidelines on how the Date/Time processing occurs:

- Relative or numeric time values are converted into absolute time values by computing the offset from the TimeBase.
- The default TimeBase is the current date. The default TimeUnit is 1 day.
- Absolute time values are converted into numeric values by computing the difference from the TimeBase in terms of the TimeUnit. This applies to both axis scale values as well as data values. For example, if the value “15 Dec 2001” is entered as data, and the TimeBase is the 20th of December, 2001, and the TimeUnit is one day, the date item will map to –5 (days) and be placed along the axis according to that value.

Parameters that can be used to input Date/Time data are:

- TopScale = (MinValue, MaxValue, StepValue);
- BottomScale = (MinValue, MaxValue, StepValue);
- LeftScale = (MinValue, MaxValue, StepValue);
- RightScale = (MinValue, MaxValue, StepValue);
- TopScroll = (ScrollMin, ScrollMax);
- BottomScroll = (ScrollMin, ScrollMax);
- LeftScroll = (ScrollMin, ScrollMax);
- RightScroll = (ScrollMin, ScrollMax);
- DataSet[1-50] = a, b, c, ...;
- LineSet[1-50] = y1, y2, y3, ...;
- LineSet[1-50] = (x1, y1), (x2, y2), (x3, y3), ...;

All other DataSet / LineSet / StockSet / BubbleSet parameters

Date/Time data can be input as absolute dates, relative time units, or numerical time units.

Absolute Date Expressions

An absolute date expression is any quoted string representing a date or time in some standard convention.
Java automatically recognizes many date/time formats, including these:

**Date Formats**
- 96/04/10
- 4/10/96
- Apr 10, 1996
- 10 Apr 96

**Time Formats**
- 12:30:00
- 12:30:00 PM
- 12:40 GMT

Use absolute date expressions for specifying the minimum or maximum values in an axis range, or when specifying an exact time-based data value.

**Example:**
- "1 Jan 1996 12:30"
- "Jan 1, 1996"
- "12:30"
- "Wed, 10 Apr 96"

This sample of chart code shows how absolute dates can be used, in context.

**Example:**
```java
BottomFormat = (DATE, "%M/%D");
BottomScale  = ("10 Apr 96", "17 Apr 96");
DataSet1     = ("10 Apr 96", 27),
              ("12 Apr 96 05:30", 50),
              ("4/13/1996", 75),
              ("4/14/96 12:30", 37),
              ("April 15, 96", 87),
              ("Apr 16, 96 12:00", 64);
```

**Relative Time Units**
You may want to specify a date or time value that is relative to another such value. Relative time units are relative to the TimeBase (described below). If the TimeBase is not defined, relative time units are relative to the current Date/Time. The relative time unit components described below may be used to input data, to set the TimeUnit attribute of the axis Format parameter, or to set the StepValue of an axis Scale parameter.

**Example:**
A measurement might be taken 20 minutes after the start of an experiment, or a task might end 5 days and 6 hours after its start. In such cases, one uses relative time units:

```
Relative Time Unit = "1Y 2M 3d 12h 30m";
```

Components are described thus:

<table>
<thead>
<tr>
<th>Time Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Years</td>
</tr>
<tr>
<td>M</td>
<td>Months</td>
</tr>
<tr>
<td>d</td>
<td>Days</td>
</tr>
</tbody>
</table>
Appendix A: Date and Time Values

Date/Time Data Format and TimeBase

Example:

"?d"       A date one week after the base date
"3d 12h 30m" A date 3 days, 12 hours, and 30 minutes after the base date and time.

The absolute date and time represented by a given relative time unit is calculated thus:

\[\text{Absolute DateTime} = \text{Axis TimeBase} + \text{Relative Time Unit}\]

Where TimeBase is defined in the Format parameter for the corresponding axis.

Numeric Time Units

Date and time values may also be specified as a numeric value, which is interpreted as follows:

\[\text{Absolute DateTime} = \text{Axis TimeBase} + N \times \text{Axis TimeUnit}\]

Where TimeBase and TimeUnit are defined in the Format parameter for the corresponding axis. This allows data values to be specified as multiples of a given time unit.

The following parameters would generate the same chart as in the code example for absolute date and time, except that they use a combination of absolute dates, relative time units and numeric time units.

Example:

```plaintext
BottomFormat = (DATE, "%M/%D", "10 Apr 96", "1d");
BottomScale = (0, "Apr 17, 1996");
TopFormat = (DATE, "%w", "10 Apr 96", "1d"); <!-- displays day of the week -->
TopScale = (0, "Apr 17, 1996");
DataSet1 = ("10 Apr 96", 27),
(\"2d 5h 30m\", 50),
(3.0, 75),
(\"4d 12h 30m\", 37),
(5, 87),
(6.5, 64);
```

Date/Time Data Format and TimeBase

Date/Time data labels are displayed using axis formatting parameters that convert numerical scale values into a date or time presentation. Additionally, the axis formatting parameters contain the attributes for setting the TimeBase and TimeUnit used in computing relative and numerical time units. The following parameters are used to configure Date Format and TimeBase information:

```plaintext
TopFormat = (FormatType, "FormatExpression", "TimeBase", "TimeUnit");
BottomFormat = (FormatType, "FormatExpression", "TimeBase", "TimeUnit");
LeftFormat = (FormatType, "FormatExpression", "TimeBase", "TimeUnit");
```
Appendix A: Date and Time Values

RightFormat = (FormatType, "FormatExpression", "TimeBase", "TimeUnit");

Regardless of how one enters the date or time as data, the display of the values, either as a tic
mark labels or as active labels, will be rendered using the FormatExpression attribute specified
in the TopFormat, BottomFormat, LeftFormat or RightFormat parameters.

For axes where data formatting is desired, the FormatType attribute is either DATE or
SIMPLEDATE. SIMPLEDATE allows developers to use the same format expressions as
described in the Java SimpleDateFormat class within the FormatExpression attribute.

SIMPLEDATE Format Expression Attributes

The following attributes are supported for the SIMPLEDATE format type:

<table>
<thead>
<tr>
<th>Field</th>
<th>Attribute</th>
<th>Sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>Year</td>
<td>yyyy</td>
<td>1999</td>
</tr>
<tr>
<td>Year</td>
<td>yy</td>
<td>99</td>
</tr>
<tr>
<td>Month</td>
<td>MMM</td>
<td>January</td>
</tr>
<tr>
<td>Month</td>
<td>MM</td>
<td>01</td>
</tr>
<tr>
<td>Month</td>
<td>M</td>
<td>1</td>
</tr>
<tr>
<td>Day of week</td>
<td>EEEE</td>
<td>Monday</td>
</tr>
<tr>
<td>Day of week</td>
<td>EE</td>
<td>Mon</td>
</tr>
<tr>
<td>Day of month</td>
<td>dd</td>
<td>01</td>
</tr>
<tr>
<td>Day of month</td>
<td>d</td>
<td>1</td>
</tr>
<tr>
<td>Hour (1-12)</td>
<td>hh</td>
<td>01</td>
</tr>
<tr>
<td>Hour (1-12)</td>
<td>h</td>
<td>1</td>
</tr>
<tr>
<td>Hour (0-23)</td>
<td>HH</td>
<td>00</td>
</tr>
<tr>
<td>Hour (0-23)</td>
<td>H</td>
<td>0</td>
</tr>
<tr>
<td>Hour (0-11)</td>
<td>KK</td>
<td>00</td>
</tr>
<tr>
<td>Hour (0-11)</td>
<td>K</td>
<td>0</td>
</tr>
<tr>
<td>Hour (1-24)</td>
<td>kk</td>
<td>01</td>
</tr>
<tr>
<td>Hour (1-24)</td>
<td>k</td>
<td>1</td>
</tr>
<tr>
<td>Minute</td>
<td>mm</td>
<td>00</td>
</tr>
<tr>
<td>Second</td>
<td>ss</td>
<td>00</td>
</tr>
<tr>
<td>Millisecond</td>
<td>SSS</td>
<td>001</td>
</tr>
<tr>
<td>AM/PM</td>
<td>a</td>
<td>AM</td>
</tr>
<tr>
<td>Time zone</td>
<td>zzzz</td>
<td>EST</td>
</tr>
<tr>
<td>Time zone</td>
<td>zz</td>
<td>ET</td>
</tr>
<tr>
<td>Day of week in month</td>
<td>F</td>
<td>3rd Thursday</td>
</tr>
<tr>
<td>Day in year</td>
<td>DDD</td>
<td>001</td>
</tr>
<tr>
<td>Day in year</td>
<td>D</td>
<td>1</td>
</tr>
<tr>
<td>Week in year</td>
<td>WW</td>
<td>1</td>
</tr>
<tr>
<td>Era</td>
<td>G</td>
<td>AD</td>
</tr>
</tbody>
</table>
Appendix A: Date and Time Values

Date/Time Data Format and TimeBase

DATE Format Expression Attributes

The following attributes are supported for the DATE format type:

<table>
<thead>
<tr>
<th>Field</th>
<th>Attribute</th>
<th>Sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>Locale form</td>
<td>%L</td>
<td>Mon Jan 01 00:00:00 2000</td>
</tr>
<tr>
<td>GMT form</td>
<td>%G</td>
<td>1 Jan 2001 00:00:00 GMT</td>
</tr>
<tr>
<td>Year</td>
<td>%Y</td>
<td>1999</td>
</tr>
<tr>
<td>Year</td>
<td>%y</td>
<td>99</td>
</tr>
<tr>
<td>Month</td>
<td>%M</td>
<td>01</td>
</tr>
<tr>
<td>Month</td>
<td>%n</td>
<td>January</td>
</tr>
<tr>
<td>Month</td>
<td>%N</td>
<td>January</td>
</tr>
<tr>
<td>Weekday</td>
<td>%W</td>
<td>Monday</td>
</tr>
<tr>
<td>Weekday</td>
<td>%w</td>
<td>Mon</td>
</tr>
<tr>
<td>Day of month</td>
<td>%d</td>
<td>01</td>
</tr>
<tr>
<td>Hour</td>
<td>%H</td>
<td>00</td>
</tr>
<tr>
<td>Minute</td>
<td>%m</td>
<td>00</td>
</tr>
<tr>
<td>Second</td>
<td>%s</td>
<td>00</td>
</tr>
</tbody>
</table>

TimeBase

The TimeBase attribute specifies the base date to be used when determining the actual date/time value when a time unit or numeric value is used. It effectively relates the 0 numeric axis value to the specified TimeBase date/time. By default, the TimeBase is set to the current Date/Time. Consider the following example. If the BottomScale is set to:

BottomScale = (-50, 50, 10);

And, if the BottomFormat is set to:

BottomFormat = (DATE, "%d/%n/%y", "1 Jan 2001", "1d");

Then the tic labels would appear as:

12/Nov/00  -50 days from 1 Jan 2001
22/Nov/00  -40 days from 1 Jan 2001
02/Dec/00  -30 days from 1 Jan 2001
12/Dec/00  -20 days from 1 Jan 2001
22/Dec/00  -10 days from 1 Jan 2001
01/Jan/01   0 days from 1 Jan 2001
11/Jan/01   +10 days from 1 Jan 2001
21/Jan/01   +20 days from 1 Jan 2001
31/Jan/01   +30 days from 1 Jan 2001
10/Feb/01   +40 days from 1 Jan 2001
20/Feb/01   +50 days from 1 Jan 2001

And similarly, if the BottomFormat is then changed to

BottomFormat = (DATE, "%d/%n/%y", "15 Jul 2001", "1d");
Then the tic labels would be changed to:

26/May/01 -50 days from 15 Jul 2001
05/Jun/01 -40 days from 15 Jul 2001
15/Jun/01 -30 days from 15 Jul 2001
25/Jul/01 -20 days from 15 Jul 2001
05/Jul/01 -10 days from 15 Jul 2001
15/Jul/01 0 days from 15 Jul 2001
25/Jul/01 +10 days from 15 Jul 2001
04/Aug/01 +20 days from 15 Jul 2001
14/Aug/01 +30 days from 15 Jul 2001
24/Aug/01 +40 days from 15 Jul 2001
03/Sep/01 +50 days from 15 Jul 2001

Please note, if you have input absolute time values as DataSet (or other Set) data, and you have explicitly set up the Scale parameter with absolute time values, then you should not use the TimeBase or TimeUnit axis Format attributes.

**TimeUnit**

The TimeUnit axis Format attribute helps to compute numeric values into date or time values. If the value is “1d” (as shown in the above example) then each discrete value in the axis is equivalent to one day. The available TimeUnit symbols are the same as those described in the Relative Time Unit section, described above.

Index

$CODEBASE, 325
$DOCBASE, 325
$ICONS, 325
$IMAGES, 325
$NETCHARTS, 325
$PATTERNS, 325
$SYMBOLS, 325

%f, 324
%g, 324

<PARAM> tag, 11
usage of, 10

A

active label
ActiveClicks, 53
dwell box, 140
dwell label, 138, 139, 141, 192, 193
scrolling, 44

title box, 51
titles, 44

Axis
  ScaleSet, 265
  Tic Layout, 301, 303
  Tics, 300
axis active labels, 43, 44
axis labels
  location in chart, 17
axis modifications
  parameters associated with, 44
axis parameters, 44

Axis Scale, 263

axis tics
  location in chart, 17
AxisThickness, 58
AxisTitleActiveLabel, 59, 90, 93

AxisZoom, 317
  BottomZoom, 317
  LeftZoom, 317
  RightZoom, 317
  RubberbandBorderStyle, 318
  RubberbandFill, 319
  specifying the color of the rubberbanded box, 319
  TopZoom, 317

Background, 51
  images in, 62, 63, 70, 91, 96, 126, 203, 270, 276, 293
  parameter definition, 62, 63, 70, 91, 96, 126, 203, 270, 276, 293
Background, 62, 63, 70, 270, 276, 293

bar chart, 21
  active labels for stacked charts, 288, 289
  appearance of bars, 65, 77, 80, 81, 109, 112, 113, 145, 210, 211, 274, 316
  axes for data sets, 114
  data values, 119
  depth of bars in, 64
  orientation of bars in, 90, 148, 153, 315
  parameters specific to a, 21, 30
  specifying a fill pattern for 3D bars, 72, 78
  specifying a fill pattern in 3D grouped stacked bar, 74, 79
  specifying active labels on bars, 64
  specifying bar animation, 65
  specifying bar corners, 67
  specifying bar drop shadows, 69
  specifying bar highlights, 71
  specifying spotlight overlays over bars, 75
  stacking in, 154, 242
Bar3DDepth, 64
BarActiveLabels, 64
BarAnimationStyle, 65
BarBorder, 65
BarCorners, 67
BarDropShadow, 69
BarHighlights, 71

BarRightFillPattern, 72
BarRightFillPattern[n][p][n], 73
BarSpotlights, 75
BarStyle, 77
BarTopFillPattern, 78
BarTopFillPattern[n][p][m], 79
BarValueLabel, 77, 80
BarValueLabelBox, 81
BarValueLabelStyle, 81
BarWidth, 81
BestFit, 82
BorderColor attribute, 320

borders, 320
BorderType attribute, 320
BorderWidth attribute, 320

bottom axis
  active labels on, 82
  color, 83, 108
  custom tic mark labels, 84
  formatting numbers on, 84
  optional labels, 89
  optional title background box, 90
  scale, 85, 132, 263, 308
  scroll bars, 87
  tic length, 87
  tic mark layout, 88
  title active labels, 82
  title box, 83

BottomActiveLabels, 82
BottomAxisTitleActiveLabel, 82
BottomAxisTitleBox, 83
BottomColor, 83, 108, 180
BottomDrawMinorTics, 83
BottomFormat, 84
BottomLabels, 84

BottomMargins, 85
BottomScale, 85
BottomScroll, 86

BottomTicLength, 87
BottomTics, 88
BottomTitle, 89
BottomTitleBox, 90

box chart, 23
  axes for data sets, 114
  box height specification, 92, 94, 95
data types, 120
data values, 119
  median, 214, 215
  outliers, 229
  parameters specific to a, 23
Boxchart
  ShowDataPoints, 274

boxes, 51
BoxBorder, 90
BoxFillPattern, 91
BoxHeight, 92
BoxLimitLines, 93
BoxLimitLineStyle, 94
BoxSymbolWidth, 94
BoxWidth, 94
bubble chart, 24
  adding data points dynamically, 55
  axes for data sets specified, 96
  bubble set scales, 98
  bubble set vectors, 100
  data sets definition, 99
  line style, 109, 145, 207
  parameters specific to a, 25
  specifying bubble animation style, 95
  symbols in, 101
BubbleAnimationStyle, 95
BubbleAxis, 96
BubbleFillPattern, 96
BubbleScale, 98
BubbleSet[n], 100
BubbleSets, 99
BubbleSymbol, 100

specifying bar animation, 65
specifying bar corners, 68
specifying bar drop shadows, 69
specifying bar highlights, 72
specifying spotlight overlays over bars, 75
stacking in, 154, 242
symbols, 111, 208
  top axis appearance, 304
Combo chart
  specifying a line drop shadow, 202
CumulativeLineSetName, 109
CumulativeLineStyle, 109
CumulativeLineSymbol, 111
CumulativeLineValueLabel, 112
CumulativeLineValueLabelBox, 113
CumulativeLineValueLabelStyle, 113

C
dashes in formatted expressions, 324
data sets
  identifying, 327
DataAxis, 114
DataPointActiveLabels, 116
DataPointColor, 117
DataPointJitter, 117
DataPointSymbol, 118
DataSet[n], 119
DataSets, 120
DataType, 120
dates, 151, 152, 181, 182, 257, 307, 330
  absolute, 331
DebugClear, 122
DebugSe, 122
descent, 48
diagram chart, 27
  lines and arrows, 142
  node box, 216
  node definition, 218
  node drag switch, 217, 223
  node label characteristics, 218
  parameters specific to a, 27
dial chart, 27
  active labels for sectors, 266
  assigning dials, 129
  assigning hands to dials, 168
  coloring dials, 127, 128
  deleting sectors, 268
dial borders, 124
dial scales, 132
dividing dials into sectors, 272
dragging dials, 167
dragging sectors, 269
hand active labels, 165
hand appearance, 170, 172, 174
hands, defining, 166
hands, deleting, 166
parameters specific to a, 28
removing dials, 125
sector appearance, 267
Index

sector divisions, 267
sector labels, 271
specifying dial hand animation style, 129
specifying dial sector animation, 133
tic labels, 134, 135, 169
tic marks, 136
DialActiveLabels, 123
DialBorders, 124
DialClip, 124
DialClipPad, 125
DialDelete, 125
DialFillPattern, 126
DialFills, 127
DialFormats, 128
DialHandAnimationStyle, 129
Dials, 129
DialScale, 131
DialSectorAnimationStyle, 133
DialSize, 134
DialSquare, 134
DialTicLabels, 134
DialTicLabelStyles, 135
DialTics, 136
DrawFences, 137
DrawOrder, 138
drill-down
  dwell label
    box, 51
    box background, 140
dwell labels, 43
DwellAnimationHighlightBorderStyle, 138
DwellAnimationHighlightFill, 139
DwellAnimationStyle, 139
DwellBox, 140
DwellLabel, 54, 140
DwellOffset, 141

E

Edges, 142
EightyLineSetName, 145
EightyTwentyLineStyle, 145
EightyTwentyLineSymbol, 146

F

FenceActiveLabels, 148
FencePosition, 148
font
  sizes, 323
FontEncoding, 149
FontName attribute, 322
FontSize attribute, 323
footer, 149
  box, 51
  box background, 150
Footer, 149
FooterActiveLabel, 150
FooterBox, 150
footers
  location in chart, 17
FormatExpr attribute, 323
frames
  targets, 327

G

GraphLayout, 153
GraphType, 153
grid, 154
  3-D depth, 155
  axes for, 156
  color, 154
  line color, 115, 155
  line style, 115, 164
  location in chart, 17
Grid, 115, 154
Grid3DDepth, 155
GridAnimationStyle, 156
GridAxis, 156
GridBlockActiveLabels, 157
GridBlockActiveLabels, 157
GridBlockSizeBackgroundColor, 157
GridBlockBackgroundColor, 157
GridBlockCellColorType, 158
GridBlockCellColorType, 158
GridBlockColors, 158
GridBlockColorSpectrum, 158
GridBlockColorSpectrum, 158
GridBlockExpressions, 159
GridBlockLabel, 159
GridBlockLabels, 160
GridBoxLayout, 161
GridBlockLine, 161
GridBlockSort, 162
GridBlockValueFormat, 163
GridBlockValueFormat, 163
GridBlockValues, 162
GridBlockValueStyle, 163
GridLine, 115, 164
grids
  3-D depth, 48
  axes, 48
defined, 47
  lines, 48
  parameters specific to, 48
GroupStackLabels, 165
GroupStackSegmentLabels, 165

H

HandActiveLabels, 165
HandData, 166
HandDelete, 166
HandDrag, 167
HandLabels, 169
Index

Hands, 168
HandStyles, 170
header
   box, 51
   box background, 172
defining a, 171
Header, 171
HeaderActiveLabel, 62, 172
HeaderView, 172
headers
   location in chart, 17
   heat map, 29
   parameters specific to a, 29
hexidecimal colors, 321
HTML documents
   replacing using active labels, 53
HTML pages
   targets in, 327
I
   identifying chart elements, 327
   ImageFormat attribute, 324
images, 62, 71, 74, 79, 80, 92, 97, 127, 154, 204, 271
   formatting in a region, 324
   URL, 325
   ImageURL attribute, 325
J
   Java font names, 323
L
   label
      angles allowed, 319
      defined as an attribute, 325
defining a, 171
Label, 175
LabelAnimationStyle, 174
LabelPos, 175
labels
   defined, 48
   in legends, 49
   on axes, 49
   parameters specific to, 49
titles as, 49
Layout, 176
leading, 48
left axis
   active label for optional title, 186
   box background for optional title, 186
   box background for title, 180
color, 180
numeric format, 181
optional custom labels, 175, 183
optional title, 185
scrolling, 184
tic label style, 187
tic length, 185
title, 179
LeftActiveLabels, 178
LeftAxis, 178
LeftAxisTitle, 179
LeftAxisTitleActiveLabel, 179
LeftAxisTitleBox, 180
LeftColor, 180
LeftFormat, 181
LeftLabels, 175, 183
LeftMargins, 183
LeftScroll, 183
LeftTicLength, 184
LeftTics, 187
LeftTitle, 185
LeftTitleActiveLabel, 186
LeftTitleBox, 186
legend
   active labels in, 188
   box background for, 190, 191
   labels, 49
   layout, 196
   location of, 189
   optional items, 194
Legend, 188
LegendActiveLabels, 188
LegendAnimationStyle, 189
LegendAxis, 189
LegendBox, 190
LegendBoxSize, 191
LegendDwellAnimationHighlightBorderStyle, 192
LegendDwellAnimationHighlightFill, 193
LegendDwellAnimationStyle, 193
LegendItems, 194
LegendLayout, 196
legends
   active labels in, 49
   boxes around, 49
defining a, 49
   layout, 49
   parameters specific to, 49
line chart, 31
   3-D lines, 199
   active labels for stacked charts, 288, 289
defining line sets, 206
   line data sets, 205
   line set axes, 201
   line style, 109, 145, 207
   line width, 211
   orientation of bar in, 90, 148, 153, 315
   parameters specific to a, 31
   specifying a line drop shadow, 202
   specifying line animation style, 200
   stacking in, 154, 242
   symbols, 111, 208
   line styles, 326
   Line3DDepth, 199
   LineAnimationStyle, 200
   LineAxis, 201
   LineDropShadow, 202
Index

LineFillPattern, 203
LineLabels[n], 204
lines, 142
LineSet[n], 205
LineSets, 206
LineStyle, 207
LineSymbol, 208
linethrough, 48
LineWidth, 211
LineWidth attribute, 326
LineValueLabel, 210
LineValueLabelBox, 210
LineValueLabelStyle, 211
meanLineAdvance, 48
MeanActiveLabels, 212
MeanColor, 213
MeanLine, 213
MeanSymbol, 214
MedianColor, 214
MetaData, 215
MinimumDataPoints, 215
multi-chart, 32
charts appearing in a, 104
defining charts with URLs, 107
layout of charts in a, 176
parameters specific to a, 32
replacement for NFParamScript, 106
sashes dividing charts, 262
multipie
specifying pie edge highlights, 233
multipie chart, 32
specifying a pie drop shadow, 232
specifying pie animation, 288
specifying pie slice animation, 275
specifying spotlight overlays over pies, 234, 238
name attribute, 326
NaturalDisplayOrder, 216
NFParamScript
defined, 11
NFParamServer, 13
NFParamURL, 12
NodeBox, 216
NodeDrag, 217, 223
NodeLabel, 218
Nodes, 218
note
active labels, 220
active labels for, 50
arrows to and from, 220
axis location in chart, 221
box background, 103, 134, 223, 244, 245, 248, 249, 251, 252
boxes, 51
defined, 50
identifying note sets, 226, 228
labels, 50
locating in charts using axes, 50
parameters specific to, 50
sets, 50
sets of notes, 224
text appearing in, 224
NoteActiveLabels[n], 220
NoteArrow, 220
NoteAxis, 221
NoteBox, 223
NoteLabel, 224
NoteSet[n], 224
NoteSets, 226
NumberFormat, 228
numeric labels
formatting expressions for, 323
 outlierActiveLabels, 228
OutlierColor, 229
OutlierSymbol, 229
overline, 48
parameter
in an include file, 12
parameter file, 12
loading parameters from a, 53
parameter script, 11
parameter server, 13
connection processing, 14
parameters
dynamically updated from another applet, 14
in dynamic charting, 11
pointing to with URL, 12
updated from JavaScript, 15
within applets, 10
Pareto
ShowEightyTwentyLines, 274
TwentyLineSetName, 315, 316, 317
Pareto chart
specifying a fill pattern for 3D bars, 72, 78
specifying a fill pattern in 3D grouped stacked bar, 74, 79
specifying bar animation, 65
specifying bar corners, 68
specifying bar drop shadows, 69
specifying bar highlights, 72
specifying spotlight overlays over bars, 75
PercentileN, 230
Pie chart, 35
3-D depth of, 240
Index

data, 276, 283
keeping the pie round, 240
label placement, 280
label position, 175
numeric data format, 278
parameters specific to a, 33, 35
pre-exploding slices, 283
slice appearance, 275, 276
slice label appearance, 279, 281, 282, 287
slice label background, 279
slice label identification, 281
slice labels, 280
specifying a pie drop shadow, 232
specifying pie animation, 288
specifying pie edge highlights, 233
specifying pie slice animation, 275
specifying spotlight overlays over pies, 234, 238
starting piece position, 230, 231, 235, 236, 237
Pie3DDepth, 240
PieAngle, 230
PieAngles, 231
PieBackgrounds, 231
PieDropShadow, 231
PieEdgeHighlights, 233
PieHighlights, 233
PieLabel, 234
PieLabelBox, 235
PieLabelLocation, 236
PieLabels, 236
PieLayout, 235
PieMarin, 236
PieSize, 237
PieSpotlights, 237
PieSquare, 240
PloarLabel, 243
PlotArea, 241
PlotType, 242
PolarLabelFormat, 244
PolarLabelStep, 244
PolarScale, 245
PolarSize, 245
PolarSquare, 245
PolyActiveLabels, 246
PolyColor, 246
PolySet, 247

R

RadarSize, 247
RadarSquare, 247
RadialAxes, 248
RadialAxesAngles, 248
RadialAxesColors, 249
RadialAxesFormat, 249
RadialAxesTitleActiveLabels, 252
RadialGrids, 252
region
background type, 320
border color, 320
defined, 51
images in a, 324
parameters specific to, 51
RelativeBoxSymbolWidth, 252
right axis
active label for optional title, 261
active labels, 253
background for optional title, 262
color, 255
custom tic mark labels, 258
numeric label formatting, 256
optional title, 261
scrolling, 258
tic length, 259
tic mark labels, 260
title, 254
title active label, 254
title background, 255
RightActiveLabels, 253
RightAxis, 253
RightAxisTitle, 254
RightAxisTitleActiveLabel, 254
RightAxisTitleBox, 255
RightColor, 255
RightDrawMinorTics, 256
RightFormat, 256
RightLabels, 258
RightMargins, 258
RightScroll, 258
RightTicLength, 259
RightTics, 256
RightTitle, 261
RightTitleActiveLabel, 261
RightTitleBox, 262
RubberbandBorderStyle, 318
RubberbandFill, 319
S
Sash, 262
ScaleFactor, 264
ScaleMode, 264
scroll bars
location in chart, 17
scrolling, 44
SectorActiveLabels[n], 266
SectorBorders, 267
SectorColors, 267
SectorData, 267
SectorDelete, 268
SectorDrag, 269
SectorLabels, 271
Sectors, 272
ShowGroupStackLabels, 274
SliceAnimationStyle, 275
SliceBorder, 275
SliceColors, 276
SliceData, 276
SliceFormat, 278
SliceLabel, 279
SliceLabelBox, 279
Index

SliceLabelContent, 280
SliceLabelContentDelimiter, 280
SliceLabelLine, 281
SliceLabels, 281
SliceLabelStyle, 282
SlicePos, 283
Slices, 283
SliceSet, 287
SliceSets, 287
SliceSlide, 288
StackDisplayOrder, 288
stacking, 154, 242
StackLabel, 288, 289
stock
  specifying bar drop shadows, 69
stock chart, 38
  3-D lines, 199
  active labels for stacked charts, 288, 289
  active labels on lines, 204
  axes for data sets, 114
  axis mapping, 290
  bar and tic display, 296
data, 291
data set display attributes, 295
data values, 119
defining line sets, 206
line data sets, 205
line set axes, 201
line style, 109, 145, 207
line width, 211
orientation of bars in, 90, 148, 153, 315
override default active labels, 294
parameters specific to a, 39
specifying a fill pattern for 3D bars, 72, 78
specifying a fill pattern in 3D grouped stacked bar, 74, 79
specifying active labels on bars, 64
specifying bar animation, 65
specifying bar corners, 68
specifying bar highlights, 72
specifying spotlight overlays over bars, 75
stacking in, 154, 242
symbols, 111, 208
Stock chart
  specifying a line drop shadow, 202
Stock Chart
  specifying stock animation, 289
StockAnimationStyle, 289
StockAxis, 290
StockColorTable, 291
StockData[n], 291
StockLabels[n], 294
StockSets, 294
StockWidth, 296
strings, 325, 327
strip chart, 40
  appending data sets, 56, 250, 299
  axes for data sets, 114
  data update, 314
  data values, 119
layout, 297
parameters specific to a, 40
StripLayout, 297
symbols
  for bubbles, 101
  for lines, 111, 208
T
Target attribute, 327
TaskColorTable, 298
TaskHeight, 299
text strings, 325, 327
Tic Locations, 299
tic marks, 44
layout, 44
time chart, 41
  axes for data sets, 114
data values, 119
parameters specific to a, 41
time values, 330
calculating numeric units, 332
calculating relative numbers, 331
title, 44
active labels for, 43, 44
boxes, 51
location in chart, 17
ToggleDataVisibility, 303
top axis
  active label for title, 305
  active labels, 303
color, 306
custom tic mark labels, 299, 308
numeric label format, 151, 306
scale, 308
scrolling, 265, 310
tic label layout, 301, 303, 312
tic length, 313
tic mark label appearance, 300, 311
title, 59, 304
title background, 60, 305
TopActiveLabels, 303
TopAxis, 304
TopAxisTitle, 59, 304
TopAxisTitleActiveLabel, 305
TopAxisTitleBox, 60, 305
TopColor, 306
TopDrawMinorTics, 306
TopFormat, 151, 306
TopLabels, 308
TopMargins, 308
TopScale, 308
TopScroll, 265, 310
TopTicLayout, 301, 303, 312
TopTicLength, 313
TopTics, 300, 311
TwentyLineSetName, 315, 316, 317

CDL Reference Manual – 7.1

© Visual Mining, Inc 2002-2011
www.visualmining.com
Index

U

underline, 48
UniqueTaskColors, 314
Update, 314
URL attribute, 328

W

WhiskerType, 315
Width attribute, 328
Windows font names, 323
WW, 324

X

XAxis attribute, 328
X-Y chart, 41

Y

YAxis attribute, 329

Z

Zoom
AxisZoom. See