

ElegantJ Indicator & Gauges

Programmer's Guide – Version 1.0.03

Table of Content

1	<u>PREFACE</u>	4
1.1	PURPOSE OF THIS DOCUMENT	4
1.2	ASSUMPTIONS	4
1.3	DOCUMENT ORGANIZATION	4
1.4	RELATED DOCUMENT	4
1.5	CONVENTIONS USED	5
2	<u>ABOUT ELEGANTJ INDICATORS & GAUGES</u>	6
2.1	PRODUCT FEATURES	6
2.1.1	DIAL GAUGE	6
2.1.2	LEVEL GAUGE	6
2.1.3	NEEDLE GAUGE	7
2.1.4	THERMOMETER GAUGE	7
2.2	TOOLKIT	7
3	<u>PRODUCT DEMO</u>	8
3.1	INSTALLING DEMO	8
3.2	VIEW DEMO	8
3.2.1	GAUGE DEMO APPLET	8
3.2.2	DIAL GAUGE DEMO APPLICATION	8
3.2.3	LEVEL GAUGE DEMO APPLICATION	9
3.2.4	NEEDLE GAUGE DEMO APPLICATION	9
3.2.5	THERMOMETER GAUGE DEMO APPLICATION	10
4	<u>INSTALLATION AND REGISTRATION</u>	11
4.1	PREREQUISITE	11
4.2	INSTALLATION PROCEDURE	11
4.2.1	INSTALLING ON WINDOWS 95/98	11
4.2.2	INSTALLING ON WINDOWS NT/2000	11
4.2.3	INSTALLING ON UNIX/LINUX	12
4.3	USING WITH DIFFERENT IDES	12
4.3.1	BORLAND JBUILDER	12
4.3.2	IBM VISUAL AGE	15
4.3.3	FORTE4JAVA	19
4.3.4	ORACLE JDEVELOPER	21
4.4	GETTING REGISTERED	23
4.4.1	HOW TO PURCHASE	23
4.4.2	USING WITH GRAPHICAL USER INTERFACE OF IDE	23
4.4.3	USING DIRECTLY WITH SOURCE CODE	23
5	<u>PROGRAMMER'S FAQ</u>	24
5.1	ELEGANTJ NUMERICAL DIAL GAUGE	24
5.1.1	HOW TO CREATE AN INSTANCE OF ELEGANTJ NUMERICDIALGAUGE?	24
5.1.2	HOW TO APPLY FORMATTING ATTRIBUTES?	25

5.1.3	HOW TO CONFIGURE CAPTION PROPERTIES?	26
5.1.4	HOW TO CONFIGURE COLOR PROPERTIES?	26
5.1.5	HOW TO CONFIGURE GAUGE VALUES PROPERTIES?	27
5.1.6	HOW TO CONFIGURE HEADER PROPERTIES?	28
5.1.7	HOW TO CONFIGURE FOOTER PROPERTIES?	29
5.1.8	HOW TO CONFIGURE UNIT BOX OPTION PROPERTIES?	29
5.1.9	HOW TO CONFIGURE GRAD OPTION PROPERTIES?	30
5.1.10	HOW TO CONFIGURE SUB GRAD OPTION PROPERTIES?	30
5.1.11	HOW TO CONFIGURE DIRECTION PROPERTIES?	31
5.1.12	HOW TO CONFIGURE MOUSE ACTIVITY TO MOVE THE NEEDLE?	31
5.1.13	HOW TO CONFIGURE KEYBOARD ENABILITY?	31
5.1.14	HOW TO CONFIGURE TEXT BOX PROPERTIES?	31
5.1.15	HOW TO CONFIGURE NEEDLE PROPERTIES?	32
5.1.16	HOW TO CONFIGURE DIAL PROPERTIES?	33
5.2	ELEGANTJ LEVEL GAUGE	33
5.2.1	HOW TO CREATE AN INSTANCE OF ELEGANTJ LEVELGAUGE?	33
5.2.2	HOW TO APPLY FORMATTING ATTRIBUTES IN ELEGANTJ LEVEL GAUGE?	34
5.2.3	HOW TO CONFIGURE CAPTION PROPERTIES IN ELEGANTJ LEVEL GAUGE?	34
5.2.4	HOW TO CONFIGURE COLOR PROPERTIES IN ELEGANTJ LEVEL GAUGE?	35
5.2.5	HOW TO CONFIGURE GAUGE VALUES PROPERTIES IN ELEGANTJ LEVEL GAUGE?	35
5.2.6	HOW TO CONFIGURE HEADER PROPERTIES IN ELEGANTJ LEVEL GAUGE?	36
5.2.7	HOW TO CONFIGURE FOOTER PROPERTIES IN ELEGANTJ LEVEL GAUGE?	37
5.2.8	HOW TO CONFIGURE TICK PROPERTIES IN ELEGANTJ LEVEL GAUGE?	37
5.2.9	HOW TO CONFIGURE MOUSE ACTIVITY TO MOVE THE INDICATION IN ELEGANTJ LEVEL GAUGE?	38
5.2.10	HOW TO CONFIGURE GAUGE PROPERTIES IN ELEGANTJ LEVEL GAUGE?	39
5.3	ELEGANTJ NEEDLE GAUGE	40
5.3.1	HOW TO CREATE AN INSTANCE OF ELEGANTJ NEEDLEGAUGE?	40
5.3.2	HOW TO APPLY FORMATTING ATTRIBUTES IN ELEGANTJ NEEDLE GAUGE?	40
5.3.3	HOW TO CONFIGURE CAPTION PROPERTIES IN ELEGANTJ NEEDLE GAUGE?	41
5.3.4	HOW TO CONFIGURE COLOR PROPERTIES IN ELEGANTJ NEEDLE GAUGE?	41
5.3.5	HOW TO CONFIGURE GAUGE VALUES PROPERTIES IN ELEGANTJ NEEDLE GAUGE?	41
5.4	ELEGANTJ THERMOMETER GAUGE	42
5.4.1	HOW TO CREATE AN INSTANCE OF ELEGANTJ THERMOMETERGAUGE?	42
5.4.2	HOW TO APPLY FORMATTING ATTRIBUTES IN ELEGANTJ THERMOMETER GAUGE?	42
5.4.3	HOW TO CONFIGURE CAPTION PROPERTIES IN ELEGANTJ THERMOMETER GAUGE?	43
5.4.4	HOW TO CONFIGURE COLOR PROPERTIES IN ELEGANTJ THERMOMETER GAUGE?	44
5.4.5	HOW TO CONFIGURE GAUGE VALUES PROPERTIES IN ELEGANTJ THERMOMETER GAUGE?	44
6	PRODUCT AND SUPPORT INFORMATION	46

1 Preface

This preface describes the document. The preface contains the following sections:

Section	Page
Error! Unknown switch argument.	4
Error! Unknown switch argument.	4
Error! Unknown switch argument.	4
Related Document	4
Conventions Used	5

1.1 Purpose of this document

The purpose of this document is to provide the fundamental skills necessary to productively install and use the ElegantJ Indicators & Gauges. This document provides both programmer and user perspective to the audience.

1.2 Assumptions

This manual assumes that readers are having reasonable level of exposure to fundamentals of Java programming and various architectures.

1.3 Document Organization

This document is organized as described in following tables.

Table 1 – Organization of the document

Chapter	Contents
Error! Unknown switch argument.	<ul style="list-style-type: none">➤ Error! Unknown switch argument.➤ Error! Unknown switch argument.
Error! Unknown switch argument.	<ul style="list-style-type: none">➤ Error! Unknown switch argument.➤ Error! Unknown switch argument.
Error! Unknown switch argument.	<ul style="list-style-type: none">➤ Error! Unknown switch argument.➤ Error! Unknown switch argument.➤ Error! Unknown switch argument.➤ Error! Unknown switch argument.
Error! Unknown switch argument.	<ul style="list-style-type: none">➤ Core components and definitions for different indicators and gauges➤ Programmers' guide to frequently asked questions

1.4 Related Document

Readers of this document can also refer to other ElegantJ Charts documents.

- [ElegantJ Charts API Documentation](#)

1.5 Conventions Used

File	Italic (slanted) type indicates variable values, instruction operands.
[] { }	In syntax definitions, brackets indicate items that are optional and braces indicate items that are required. Vertical bars separating items inside brackets or braces indicate that you choose one item from among those listed.
. . .	In syntax definitions, a horizontal ellipsis indicates that the preceding item can be repeated one or more times.
//	An explanation of a particular function performed by preceding code.

2 About ElegantJ Indicators & Gauges

When monitoring of data changes is very important

Show live stock price changes in stock market on your PC screen and get alert when it changes by 5 %. Monitor temperature changes and heat control, monitor CPU usage of your server or pressure in a tank and get alert when it reaches dangerous mark. Get data from centrifuges, heating systems, or pressure systems and monitor it in real time.

Get alerts and prompts when they reach a specific level. You can specify levels when you want to get alerts and prompts. When specified levels are reached, change color properties of indicator itself. Alternatively, ElegantJ Gauges Bean will give you an event, which you can use to interface with your external application to carryout actions like sending a pager message, SMS message or an e-mail message on receipt or closing a valve.

Your JAVA based solution can have all of these using ElegantJ Gauges bean. Want to show levels, needle or dial, we have beans for your indicator needs. Also, Configurable visual appeal, colors and other properties so that output fits appropriately in your solution. Free yourself from the stress of designing gauges and catching alerts.

2.1 Product Features

2.1.1 Dial Gauge

- Edges of dial is now sharper giving better appearance.
- Configurable properties for header and footer, Unit box, Text Box, Needle and Center Circle
- You can set value on gauge using navigation keys of keyboard (Up, Down, Page Up, Page Down, Home and End) and also by a mouse-click on the dial itself.
- Events for you to trap - valueChanged, zoneChanged
- Option to move in clock-wise direction and anti-clock wise direction
- Tick Properties - type, length, width, color, gap
- Values options - Fraction, font, font color, distance
- Zone Properties - name and depth
- Dial properties - radius and border
- Gauge start angle and end angle

2.1.2 Level Gauge

- Configurable properties for header and footer, unit box
- Configurable properties for Unit box
- Events for you to trap - valueChanged, zoneChanged
- You can set value on gauge by a mouse-click on the dial itself.
- Tick Properties - type, length, width, color, gap
- Values options - Fraction, font, font color, distance
- Directions - up and down in addition to existing
- Gauge Properties - type, orientation and border
- Normal zone colors

2.1.3 Needle Gauge

- Many appearance and formatting controls like Color of background, text, font. etc.
- ElegantJ Needle Gauge JAVA bean supports two attributes for caption - Text and Font.
- In this JAVA Bean, you can customize panel's color.
- Values that you can configure - Maximum value, Minimum value, Zone values, Unit to the values, Needle value, Grad scale.
- You can set font for numbers.

2.1.4 Thermometer Gauge

- Configurable properties for header and footer, Unit box
- Events for you to trap - valueChanged, zoneChanged
- You can set value on gauge by a mouse-click on the dial itself
- Tick Properties - type, length, width, color, gap
- Values options - Fraction, font, font color, distance
- Gauge Properties - Border, color of indicator

2.2 Toolkit

ElegantJ Gauges is shipped with following Beans -

- ElegantJ NumericDialGauge (dialguage.jar)
- ElegantJ LevelGauge (levelguage.jar)
- ElegantJ NeedleGauge (needlegauge.jar)
- ElegantJ ThermometerGauge (thermometergauge.jar)

3 Product Demo

3.1 Installing Demo

Extracted file (archive) contains demos directory. All the demos are located in this demos directory.

Your classpath environment variables must be set correctly in order to run the demos. For windows users we have provided rundemo.bat file, which will run demo directly. All Demo Applets contain HTML file to run an Applet associated with it.

For more information on how to set classpath, refer Installation procedure.

3.2 View Demo

3.2.1 Gauge demo applet

This demo applet demonstrates all the ElegantJ Gauges including dial gauge, level gauge, needle gauge and thermometer gauge.

3.2.2 Dial Gauge demo application

This demo application demonstrates the use of ElegantJ NumericDialGauge JAVA Bean.

To run this demo application, go to ElegantJGauges\demos\application\dial directory, and then execute `java -jar dialgaugeapp.jar` command.

3.2.3 Level Gauge demo application

This demo application demonstrates the use of ElegantJ LevelGauge JAVABean.

To run this demo application, go to ElegantJGauges\demos\application\level directory, and then execute `java -jar levelgaugeapp.jar` command.

3.2.4 Needle Gauge demo application

This demo application demonstrates the use of ElegantJ NeedleGauge JAVA Bean.

To run this demo application, go to ElegantJGauges\demos\application\needle directory, and then execute `java -jar needlegaugeapp.jar` command.

3.2.5 Thermometer Gauge demo application

This demo application demonstrates the use of ElegantJ ThermometerGauge JAVABean.

To run this demo application, go to ElegantJGauges\demos\applications\thermo directory, and then execute `java -jar thremometergaugeapp.jar` command.

Note: For windows users we have provided the `rundemo.bat` file to run respective demo application.

4 Installation and Registration

4.1 Prerequisite

Before installing ElegantJ Beans, please ensure that your computer system and development environment are setup and working as per expectations. Person evaluating ElegantJ Beans is expected to be able to write and execute simple JAVA applications.

We provide free pre-purchase technical support to help you complete process of evaluation. Mail to support@elegantJBeans.com for pre-purchase technical support.

To be able to use ElegantJ Beans, your computer systems are required to have -

- An IDE that supports Java 2 or higher, or
- JDK 1.2.2 or higher

Please note that evaluation version of ElegantJ Beans cannot be deployed for commercial, non-commercial or any other purpose in any possible way. You may not build any software, applets or applications for distribution with evaluation version of ElegantJ Beans.

This evaluation version is to be solely used by you to evaluate suitability of ElegantJ Beans for your needs. For more details please go through License Agreement.

4.2 Installation Procedure

On extracting EJGauges.zip, following directories will be created -

- Directory [jars] contains Jar file(s)
- Directory [demos] contains demo application(s) with source code
- Directory [docs] contains User Guide and Java API

4.2.1 Installing on Windows 95/98

To include ElegantJ Beanery in CLASSPATH, add following statement to your autoexec.bat file,

```
set CLASSPATH= %CLASSPATH%; C:\ELEGANTJ\<ELEGANTJ_BENARY_HOME>  
\jars\<ELEGANTJ_BEAN>.jar;
```

Restart Windows to make changes effective.

4.2.2 Installing on Windows NT/2000

Go to Control Panel and select System. You will find environment variables on Environment tab in Windows NT systems and on Advanced tab in Windows 2000 systems. Find CLASSPATH environment variable or create it.

To include ElegantJ Beanary in the CLASSPATH, specify or add following value for variable -

```
[EXISTING-CLASSES]; C:\<ELEGANTJ_BEANARY_HOME>\jars\  
<ELEGANTJ_BEAN>.jar
```

4.2.3 Installing on UNIX/Linux

Before you begin using ElegantJ Beanary, you must manually configure CLASSPATH environment variable. CLASSPATH must point to the location of classes and installation directory. For example, to set CLASSPATH for ElegantJ Beanary,

```
setenv CLASSPATH:/usr/local/<ELEGANTJ_BEANARY_HOME>/jars/  
<ELEGANTJ_BEAN>.jar;
```

If you are using Bourne Shell, commands are,

```
CLASSPATH= $CLASSPATH:./usr/local/<ELEGANTJ_BEANARY_HOME>/jars/  
<ELEGANTJ_BEAN>.jar:export CLASSPATH
```

4.3 Using with Different IDEs

4.3.1 Borland JBuilder

Follow the steps given below to create a new category and install Bean into Borland JBuilder.

Create a New Category

- Click menu **Tools > Configure Palette**. It opens **Palette Properties** dialog box
- Click **Add** button. It opens **Add Page** dialog box. Specify name (for example ElegantJ) and click **OK**. New category gets created.

Create New Library and Install/Import Bean

- On **Palette Properties** dialog box, click **Pages** tab. Select newly created page (for example ElegantJ).

- Click button **Select Library**. **Select a Different Library** dialog box opens.

- Click **New** button on **Select a Different Library** dialog box. **New Library Wizard** dialog box opens.
- Specify name for library and click **Add** button. **Select One or More Directory** dialog box opens.

- Select directory where jar files are located, or select jar file to be imported. Click **OK**. Dialog box gets closed and **New Library Wizard** dialog box appears in front.

- Click **OK** on **New Library Wizard**. It opens **Select a Different Library** dialog box.
- Confirm that newly created library is selected. Click **OK**. It opens **Add Components** under **Palette Properties**.
- Click **Add** from **Selected Libraries** button. It opens **Results** dialog box, click **OK**.
- Results dialog box disappears. Click **OK** on **Palette Properties** dialog box.
- Beans are ready to use.

4.3.2 IBM Visual Age

Follow the steps given below to create an application in VisualAge and import ElegantJ Bean into the application.

Create an application

Open IBM VisualAge to work as Administrator.

- Click menu **File>QuickStart**. It opens **Quick Start** dialog box.

- On Left pane, click **Basic**. On right pane, click **Create Application**. Click **OK**. It opens **Create Application** step of **SmartGuide** dialog box.
- In **Project** entry box, specify **Project** (for example, ElegantJDemoApp). In **Package** entry box, specify package name (for example, ejdemoapp). In **Class** entry box, specify Class name (for example ElegantJDemoApp).
- Select "Create Swing based application" if you are working with JFC. Select "Create AWT based application" if you are working with AWT.

- Click **Finish** button. Application will be created and it will open new window for ElegantJDemoApp.

To Import Beans in the Application

- In ElegantJDemoApp (the example used here) application window, click menu **File > Import...** . **Import** dialog of **SmartGuide** opens.

- Choose **Jar file** as import source. Click **Next** to proceed.
- From **Jar/Zip file** dialog box, browse location and get jar file in **Filename** entry box. On the same dialog box, confirm that **.class** and **resources** options are selected under **What type of files do you want to import?**

- Click **Finish** button. It opens **Modify Palette** dialog box.
- To create a new category, click **New Category** button. Specify name of category (for example, ElegantJ). Confirm that newly created category is highlighted, select bean you want to import from **Available Beans** area of the dialog box.
- Click **Add To Category** button to complete the process. Click **OK**.

- Beans are ready to use.

Note: Microsoft, IBM, VisualJ, PowerJ, JBuilder, Borland, Forte, NetBeans, VisualAge, JDeveloper, SUN Microsystems, ORACLE, JAVA, WINDOWS, Internet Explorer, Netscape, Opera, Adobe, PDF are not intended to indicate any specific relation to Elegant MicroWeb Technologies Pvt. Ltd. or ElegantJ or ElegantJ Beans. The corporation who own these trademarks are not related or connected with Elegant MicroWeb Technologies Pvt Ltd or ElegantJ or ElegantJ Beans in any way. The respective corporations own these trademarks. They have only been mentioned in this and other documentations to demonstrate or support the use of these products with ElegantJ Beans or as related information. ALL TRADEMARKS ARE DULY ACKNOWLEDGED.

4.3.3 Forte4Java

Follow the steps given below to create a new category for ElegantJ Beans and add Beans to Sun Forte4Java.

Create a new Palette Category

- To create a new Palette Category, right-click the mouse on Palette Categories tab. A context menu opens.
- Click menu option Create New Category. It opens New Palette Category dialog box. Specify Category name, for example - ElegantJ. Click OK. New category is created.

To import Beans

- Click menu **Tools > Install New JavaBean...** . It opens **Install JavaBean** dialog Box.
- Browse and select the jar file that you want to install/import.
- Click **OK**. It auto-detects beans available in Jar file and displays their names in **Select JavaBean** dialog box.
- Select the bean that you want to install.
- Click **OK**. It opens **Palette Category** dialog box, which lists newly added Category.

To add selected Beans into newly created category

- From **Palette Category** dialog box, select newly created category (example here, ElegantJ)
- Click **OK**. Beans will get installed under this category.

- Beans are ready to use.

Note: Microsoft, IBM, VisualJ, PowerJ, JBuilder, Borland, Forte, NetBeans, VisualAge, JDeveloper, SUN Microsystems, ORACLE, JAVA, WINDOWS, Internet Explorer, Netscape, Opera, Adobe, PDF are not intended to indicate any specific relation to Elegant MicroWeb Technologies Pvt. Ltd. or ElegantJ or ElegantJ Beans. The corporation who own these trademarks are not related or connected with Elegant MicroWeb Technologies Pvt Ltd or ElegantJ or ElegantJ Beans in any way. The respective corporations own

these trademarks. They have only been mentioned in this and other documentations to demonstrate or support the use of these products with ElegantJ Beans or as related information. ALL TRADEMARKS ARE DULY ACKNOWLEDGED.

4.3.4 Oracle JDeveloper

Follow the steps given below to import ElegantJ Beans in Oracle JDeveloper.

To Import beans into a project

- Click menu Project > Project Settings. It opens Project Settings dialog box.

- Click **Libraries** entry appearing on the pane on left side.
- Click **New** button. It opens **New Library** dialog box.

- In **Library Name** entry box, specify Library name. In **Location** entry box, specify location where you want the library to be located. In **Class Path** entry box, specify path of jar file of Bean. Browse the location by clicking **Edit...** button if required.
- Click **OK**. The library you selected appears in **Available Libraries** list of **Project Properties** dialog box.
- Select the library you want and click **>** button to add it to **Selected Libraries** list.
- Click menu **Tools > Configure Component Palette**. It opens **Configure Component Palette** dialog box.
- To create a new palette page, click **New Page...** button. It opens **New Palette Page** dialog box. Specify **Page Name** and **Page Type**. Click **OK**. A new page with specified name will be created.

To add component to page

- On **Configure Component** dialog box, from **Pages** list, select the page and click **Add Component** button. It opens **Add JavaBeans** dialog box with available libraries.
- Select a library from **Library** combo box.
- Browse the hierarchy and select a component. It displays preview of icon for selected component. Click **OK**. It shows **Palette Confirmation** dialog box.
- Click **Yes** on **Palette Confirmation** dialog box to complete the process.
- Beans are ready to use.

Note: Microsoft, IBM, VisualJ, PowerJ, JBuilder, Borland, Forte, NetBeans, VisualAge, JDeveloper, SUN Microsystems, ORACLE, JAVA, WINDOWS, Internet Explorer, Netscape, Opera, Adobe, PDF are not intended to indicate any specific relation to Elegant MicroWeb Technologies Pvt. Ltd. or EllegantJ or ElegantJ Beans. The corporation who own these trademarks are not related or connected with Elegant MicroWeb Technologies Pvt Ltd or ElegantJ or ElegantJ Beans in any way. The respective corporations own these trademarks. They have only been mentioned in this and other

documentations to demonstrate or support the use of these products with ElegantJ Beans or as related information. ALL TRADEMARKS ARE DULY ACKNOWLEDGED.

4.4 Getting Registered

4.4.1 How to Purchase

You can purchase a license from our web site www.elegantJBeans.com
You can also contact our sales team sales@elegantJBeans.com

On purchasing a license, you will receive a mail having your license key.

4.4.2 Using with Graphical User Interface of IDE

While designing an application or an applet graphically in an Integrated Development Environment, you will find a property named LicenseKey along with other properties of the bean. Specify key (serial number) in LicenseKey property.

Note: Instead of typing the license key, we suggest you to copy the license key from mail and paste it at required place. Make sure that the selection does not have any leading or trailing spaces.

4.4.3 Using directly with source code

When you use the Bean within code, you need to set the key by providing value in setLicenseKey Method.

Note: Instead of typing the license key, we suggest you to copy the license key from mail and paste it at required place. Make sure that the selection does not have leading or trailing spaces.

```
ClassName object = new ClassName();  
object.setLicenseKey("LICENSE_KEY");
```

Example

```
AWTTree awtTree = new AWTTree();  
awtTree.setLicenseKey("LICENSE_KEY");
```

5 Programmer's FAQ

5.1 ElegantJ Numerical Dial Gauge

5.1.1 How to create an instance of ElegantJ NumericDialGauge?

ElegantJ Numeric Dial Gauge can be created in following way,

First way:

```
com.elegantj.gauges.dial.NumericDialGauge numericDialGauge = new  
com.elegantj.gauges.dial.NumericDialGauge();
```

This creates numeric dial gauge with default constructor.

Second way:

```
String caption = "ElegantJ Numerical Dial Gauge";  
int size = 320; // dial's diameter, default is 320  
com.elegantj.gauges.dial.NumericDialGauge numericDialGauge = new  
com.elegantj.gauges.dial.NumericDialGauge(caption, size);
```

This creates numeric dial gauge with specified caption and size.

Third way:

```
String caption = "ElegantJ Numerical Dial Gauge";  
int size = 320; // dial's diameter, default is 320  
boolean clockWise = true;  
com.elegantj.gauges.dial.NumericDialGauge numericDialGauge = new  
com.elegantj.gauges.dial.NumericDialGauge(caption, size, clockWise);
```

This creates numeric dial gauge with specified caption, size and direction.

Fourth way:

```
String caption = "ElegantJ Numerical Dial Gauge";  
int size = 320; // dial's diameter, default is 320  
double min = 0; // dial's minimum value, default is 0  
double max = 100; // dial's maximum value, default is 100  
com.elegantj.gauges.dial.NumericDialGauge numericDialGauge = new  
com.elegantj.gauges.dial.NumericDialGauge(caption, size, min, max);
```

This creates numeric dial gauge with specified caption, size and with min and max values.

Fifth way:

```
String caption = "ElegantJ Numerical Dial Gauge";  
int size = 320; // dial's diameter, default is 320  
boolean clockWise = true;  
double min = 0; // dial's minimum value, default is 0  
double max = 100; // dial's maximum value, default is 100  
com.elegantj.gauges.dial.NumericDialGauge numericDialGauge = new  
com.elegantj.gauges.dial.NumericDialGauge(caption, size, min, max, clockWise);
```

This creates numeric dial gauge with specified caption, size, direction and with min and max values.

Sixth way:

```
String caption = "ElegantJ Numeric Dial Gauge";
int size = 320; // dial's diameter, default is 320
boolean clockWise = true;
double min = 0; // dial's minimum value, default is 0
double max = 100; // dial's maximum value, default is 100
double minAngle = 0; // dial's minimum angle, default is 0
double maxAngle = 360; // dial's maximum angle, default is 360
com.elegantj.gauges.dial.NumericDialGauge numericDialGauge = new
com.elegantj.gauges.dial.NumericDialGauge(caption, size, min, max, minAngle,
MaxAngle, clockWise);
```

This creates numeric dial gauge with specified caption, size, direction, min - max values and min - max angles.

5.1.2 How to apply formatting attributes?

ElegantJ Numeric Dial Gauge provides the following general formatting properties,

- Font
- Background
- Foreground
- Visible
- Enabled
- Locale
- Cursor
- Bounds

- To set Font
`numericDialGauge.setFont(new java.awt.Font("Dialog", java.awt.Font.PLAIN, 12));` // sets font as dialog, style as plain, size as 12

- To set Background color
`numericDialGauge.setBackground(java.awt.Color.cyan);` // sets background color as cyan

- To set Foreground
`numericDialGauge.setForeground(java.awt.Color.black);` // sets foreground color as black

- To set Visibility
`numericDialGauge.setVisible(true);` // sets visibility as true

- To set Enability
`numericDialGauge.setEnabled(true);` // sets enability as true

- To set Locale
`numericDialGauge.setLocale(java.util.Locale.ENGLISH);` // sets locale as English

- To set Cursor
`numericDialGauge.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));` // sets cursor as hand cursor

- To set Bounds

```
int x = 5;
int y = 5;
int width = 300;
int height = 200;
numericDialGauge.setBounds(x, y, width, height); // sets bounds to x, y,
width and height
```

5.1.3 How to configure caption properties?

ElegantJ NumericDialGauge allows to customize caption property,

- To set caption

```
String caption = "ElegantJ Numeric Dial Gauge";
numericDialGauge.setCaption(caption); // sets caption
```

5.1.4 How to configure color properties?

Color for following aspects of ElegantJ NumericDialGauge can be customized,

- Disabled area's color
- Display box's color
- Text's color
- Needle's color
- Grad's color
- Subgrad's color
- InnerDial's color

- To set disabled area color

```
java.awt.Color disabledAreaColor = java.awt.Color.gray;
numericDialGauge.setDisabledAreaColor(disabledAreaColor); // sets
disabled area color as gray
```

- To set box color

```
java.awt.Color boxColor = java.awt.Color.green;
numericDialGauge.setDisplyBoxColor(boxColor); // sets display box color
as green
```

- To set text color

```
java.awt.Color displyTextColor = java.awt.Color.blue;
numericDialGauge.setDisplyTextColor(displyTextColor); // sets display text
color as blue
```

- To set needle color

```
java.awt.Color needleColor = java.awt.Color.cyan;
numericDialGauge.setNeedleColor(needleColor); // sets needle color as
cyan
```

- To set grad color

```
java.awt.Color gradColor = java.awt.Color.red;
numericDialGauge.setGradColor(gradColor); // sets grad color as red
```

- To set subgrad color

```
java.awt.Color subGradColor = java.awt.Color.black;
numericDialGauge.setSubGradColor(subGradColor); // sets sub grad color
as black
```

- To set inner dial's color

```
java.awt.Color innerDialColor = java.awt.Color.white;
numericDialGauge.setInnerDialColor(innerDialColor); // sets the inner dial
color as white
```

5.1.5 How to configure gauge values properties?

ElegantJ NumericDialGauge allows to customize following gauge values properties:

- To set current value of gauge

```
int currentValue = 50;
numericDialGauge.setValue(currentValue); // sets current value of gauge
as 50
```

- To set maximum value

```
double maximumValue = 150;
numericDialGauge.setMaximum(maximumValue); // sets maximum value
as 150
```

- To set minimum value

```
double minimumValue = 0;
numericDialGauge.setMinimum(minimumValue); // sets minimum value as
0
```

- To set new zone value

```
java.util.Vector zoneValues = new java.util.Vector();
java.util.Vector v1 = new java.util.Vector();

v1.addElement(new com.elegantj.gauges.MeterZone("Safe
Zone",0.0,60.0,Color.cyan,10,0,15));
v1.addElement(new java.lang.Double(0));
v1.addElement(new java.lang.Double(100));
zoneValues.addElement(v1);
```

```
java.util.Vector v2 = new java.util.Vector();
```

```
v2.addElement(new com.elegantj.gauges.MeterZone("Warning
Zone",60.0,80.0,Color.blue,5,4,15));
v2.addElement(new java.lang.Double(0));
v2.addElement(new java.lang.Double(100));
zoneValues.addElement(v2);
```

```
java.util.Vector v3 = new java.util.Vector();
v3.addElement(new com.elegantj.gauges.MeterZone("Alarm
Zone",80.0,100.0,Color.red,5,4,15));
v3.addElement(new java.lang.Double(0));
v3.addElement(new java.lang.Double(100));
zoneValues.addElement(v3);
```

```
dialGauge.setNewZoneValues(zoneValues);
```

- To set length of fraction part of value

```
int fractionLength = 2;
numericDialGauge.setFractionLength(fractionValue); // sets length of
fraction part of value as 2
```

- To set radius of the values to be displayed

```
double valueRadius = 5;
numericDialGauge.setValueRadius(valueRadius); // sets radius of the
values to be displayed as 5
```

- To set maximum angle

```
double maxAngle = 320;
numericDialGauge.setMaximumAngle(maxAngle); // sets maximum angle
as 320
```
- To set minimum angle

```
double minAngle = 0;
numericDialGauge.setMinimumAngle(minAngle); // sets minimum angle as
0
```

5.1.6 How to configure header properties?

ElegantJ Numeric Dial Gauge provides following header attributes:

- Header text
- Header background
- Header foreground
- Header font
- Header spacing (spacing above and below header text)
- Header visibility

- To set header text

```
String headerText = "Header";
numericDialGauge.setHeaderText(headerText); // sets header text
```
- To set header background

```
java.awt.Color backgroundColor = java.awt.Color.blue;
numericDialGauge.setHeaderBackground(backgroundColor); // sets header
background color as blue
```
- To set header foreground

```
java.awt.Color foregroundColor = java.awt.Color.white;
numericDialGauge.setHeaderForeground(foregroundColor); // sets header
foreground color as white
```
- To set header font

```
numericDialGauge.setHeaderFont(new java.awt.Font("Dialog",
java.awt.Font.PLAIN, 12)); // sets header font as dialog, plain and of size
12
```
- To set header spacing (spacing above and below header text)

```
int headerSpacing = 10;
numericDialGauge.setHeaderSpacing(headerSpacing); // sets header
spacing (spacing above and below header text) as 10 pixels
```
- To set header visibility

```
numericDialGauge.setHeaderVisible(true); // sets header visibility as true
```

5.1.7 How to configure footer properties?

ElegantJ Numeric Dial Gauge provides following footer options.

- Footer text
- Footer background
- Footer foreground
- Footer font
- Footer spacing (spacing above and below footer text)
- Footer visibility

- To set footer text

```
String headerText = "Footer";  
numericDialGauge.setFooterText(headerText); // sets footer text
```

- To set footer background

```
java.awt.Color backgroundColor = java.awt.Color.blue;  
numericDialGauge.setFooterBackground(backgroundColor); // sets footer  
background color as blue
```

- To set footer foreground

```
java.awt.Color foregroundColor = java.awt.Color.white;  
numericDialGauge.setFooterForeground(foregroundColor); // sets footer  
foreground color as white
```

- To set footer font

```
numericDialGauge.setFooterFont(new java.awt.Font("Dialog",  
java.awt.Font.PLAIN, 12)); // sets footer font as dialog, plain and of size  
12
```

- To set footer spacing (spacing above and below footer text)

```
int headerSpacing = 10;  
numericDialGauge.setFooterSpacing(headerSpacing); // sets footer  
spacing (spacing above and below footer text) as 10 pixels
```

- To set footer visibility

```
numericDialGauge.setFooterVisible(true); // sets footer visibility as true
```

5.1.8 How to configure unit box option properties?

ElegantJ Numeric Dial Gauge provides following option for unit box:

- To set text of unit

```
String text = "Km/hr";  
numericDialGauge.setUnit(text); // sets unit text
```

- To set location

```
java.awt.Point pt = new Point(100, 25);  
numericDialGauge.setUnitLocation(pt); // sets location for unit text
```

- To set font

```
java.awt.Font unitFont = new java.awt.Font("Dialog",  
java.awt.Font.PLAIN, 12);  
numericDialGauge.setUnitFont(unitFont); // sets unit font as dialog, plain  
and of size 12
```

- To set color

```
java.awt.Color unitColor = java.awt.Color.black;
```

```
numericDialGauge.setUnitColor(unitColor); // sets unit color as black
```

5.1.9 How to configure grad option properties?

ElegantJ Numeric Dial Gauge provides following grad options properties:

- The possible values for grad type are
 - `com.elegantj.gauges.dial.NumericDialGauge.CIRCULAR_GRAD`
 - `com.elegantj.gauges.dial.NumericDialGauge.LINEAR_GRAD`
- To set grad type
`numericDialGauge.setGradType(com.elegantj.gauges.dial.NumericDialGauge.CIRCULAR_GRAD);`

This sets grad type as CIRCULAR_GRAD

- To set grad length
`double gradLength = 100;`
`numericDialGauge.setGradLength(gradLength); // sets grad length as 100 pixels`
- To set grad width
`double gradWidth = 10;`
`numericDialGauge.setGradWidth(gradWidth); // sets grad width as 10 pixels`
- To set grad color
`java.awt.Color gradColor = java.awt.Color.black;`
`numericDialGauge.setGradColor(gradColor); // sets grad color as black`

5.1.10 How to configure sub grad option properties?

ElegantJ NumericDialGauge provides following sub grad option properties

- The possible values for sub grad type are
 - `com.elegantj.gauges.dial.NumericDialGauge.CIRCULAR_GRAD`
 - `com.elegantj.gauges.dial.NumericDialGauge.LINEAR_GRAD`
- To set sub grad type
`numericDialGauge.setSubGradType(com.elegantj.gauges.dial.NumericDialGauge.CIRCULAR_GRAD); // sets sub grad type as CIRCULAR_GRAD`
- To set sub grad length
`double subGradLength = 50;`
`numericDialGauge.setSubGradLength(subGradLength); // sets sub grad length as 50 pixels`
- To set sub grad width
`double subGradWidth = 10;`
`numericDialGauge.setSubGradWidth(subGradWidth); // sets sub grad width as 10 pixels`
- To set sub grad color
`java.awt.Color subGradColor = java.awt.Color.black;`
`numericDialGauge.setSubGradColor(subGradColor); // sets sub grad color as black`

5.1.11 How to configure direction properties?

ElegantJ Numeric Dial Gauge supports both clockwise as well as anticlockwise directions,

- To set direction
`numericDialGauge.setClockwise(true);` // sets direction to clockwise direction

5.1.12 How to configure mouse activity to move the needle?

ElegantJ NumericDialGauge allows to set needle value on mouse drag or press.

The possible values for the mouse activity are

- `com.elegantj.gauges.dial.NumericDialGauge.MOVE_ON_MOUSE_DRAG` - Needle value can be changed on mouse drag
- `com.elegantj.gauges.dial.NumericDialGauge.MOVE_ON_MOUSE_PRESS` - Needle value can be changed on mouse-click
- `com.elegantj.gauges.dial.NumericDialGauge.MOVE_OFF` - User can not interact with gauge on mouse event
- To set mouse activity,
`numericDialGauge.setActiveNeedleOn(com.elegantj.gauges.dial.NumericDialGauge.MOVE_ON_MOUSE_DRAG);`

This allows to set needle value on mouse drag.

5.1.13 How to configure keyboard enablement?

ElegantJ NumericDialGauge allows user to increase / decrease value of the needle on keyboard input.

- To enable keyboard activity
`numericDialGauge.setKeyboardEnabled(true);` // allows to set needle value on keyboard input

5.1.14 How to configure text box properties?

ElegantJ NumericDialGauge provides following text box properties:

- Display text
- Display text font
- Display text color
- Display box color
- Display box border
- Display box visibility
- To set text
`java.lang.String text = "Numeric Dial Gauge";`
`numericDialGauge.setDisplayText(text);` // sets display text
- To set display text font
`numericDialGauge.setDisplayTextFont(new java.awt.Font("Dialog", java.awt.Font.PLAIN, 12));` // sets display text font as dialog, plain and of size 12

- To set display text color
`numericDialGauge.setDisplayTextColor(java.awt.Color.black); // sets display text color as black`
- To set background
`numericDialGauge.setDisplayBoxColor(java.awt.Color.blue); // sets display box color as blue`
- To set display box border
 - Permissible values for border types are,
 - `com.elegantj.gauges.GaugeBorder.NONE`
 - `com.elegantj.gauges.GaugeBorder.SIMPLE`
 - `com.elegantj.gauges.GaugeBorder.RAISED`
 - `com.elegantj.gauges.GaugeBorder.LOWERED`

```
int borderType = com.elegantj.gauges.GaugeBorder.SIMPLE;
int borderWidth = 4;
java.awt.Color borderColor = java.awt.Color.green;
com.elegantj.gauges.GaugeBorder gaugeBorder =
new com.elegantj.gauges.GaugeBorder(borderType, borderWidth,
borderColor);
numericDialGauge.setDisplayBoxBorder(border); // sets display box border
```
- To set display box visibility
`numericDialGauge.setDisplayBoxVisible(true); // sets display box visibility as true`

5.1.15 How to configure needle properties?

ElegantJ NumericDialGauge provides following needle attributes:

- Needle type
- Needle color
- Needle length
- Needle width
- Needle tail length
- To set needle type
 - The possible values for needle types are
 - `com.elegantj.gauges.dial.NumericDialGauge.ARROW_NEEDLE`
 - `com.elegantj.gauges.dial.NumericDialGauge.LINEAR_NEEDLE`
 - `com.elegantj.gauges.dial.NumericDialGauge.LINEAR_CIRCLE_NEEDLE`
 - `com.elegantj.gauges.dial.NumericDialGauge.LINEAR_ARROW_NEEDLE`
 - `com.elegantj.gauges.dial.NumericDialGauge.BALLS_NEEDLE`

```
numericDialGauge.setNeedleType(com.elegantj.gauges.dial.NumericDialGauge.ARROW_NEEDLE); // sets needle type as ARROW_NEEDLE
```
- To set needle color
`numericDialGauge.setNeedleColor(java.awt.Color.black); // set needle color as black`
- To set needle length
`double needleLength = 50;`

```
numericDialGauge.setNeedleLength(needleLength); // sets needle length
as 50 pixels
```

- To set needle width

```
double NeedleWidth = 10;
numericDialGauge.setNeedleWidth(NeedleWidth); // sets needle width as
10 pixels
```
- To set needle tail length

```
double needleTailLength = 10;
numericDialGauge.setNeedleTail(needleTailLength); // sets needle tail
length as 10 pixels
```

5.1.16 How to configure dial properties?

ElegantJ NumericDialGauge provides following dial properties:

- To set dial radius

```
double dialRadius = 50;
numericDialGauge.setDialRadius(dialRadius); // sets dial radius as 50
pixels
```
- To set dial outer border
 - Permissible values for border types are,
 - `com.elegantj.gauges.GaugeBorder.NONE`
 - `com.elegantj.gauges.GaugeBorder.SIMPLE`
 - `com.elegantj.gauges.GaugeBorder.RAISED`
 - `com.elegantj.gauges.GaugeBorder.LOWERED`

```
int borderType = com.elegantj.gauges.GaugeBorder.SIMPLE;
int borderWidth = 4;
java.awt.Color borderColor = java.awt.Color.green;
com.elegantj.gauges.GaugeBorder gaugeBorder = new
com.elegantj.gauges.GaugeBorder(borderType, borderWidth,
borderColor);
numericDialGauge.setOuterDialBorder(border); // sets dial outer border
```

5.2 ElegantJ Level Gauge

5.2.1 How to create an instance of ElegantJ LevelGauge?

To create an instance of ElegantJ LevelGauge with default constructor,

```
com.elegantj.gauges.level.LevelGauge levelGauge = new
com.elegantj.gauges.level.LevelGauge();
```

This creates an instance of ElegantJ LevelGauge

5.2.2 How to apply formatting attributes in ElegantJ Level Gauge?

ElegantJ LevelGauge provides following formatting attributes,

- Font
- Background
- Foreground
- Visible
- Enabled
- Locale
- Cursor
- Bounds

- To set Font
`levelGauge.setFont(new java.awt.Font("Dialog", java.awt.Font.PLAIN, 12)); // sets fonts to dialog, style to plain, size to 12`

- To set Background
`levelGauge.setBackground(java.awt.Color.cyan); // sets background color as cyan`

- To set Foreground
`levelGauge.setForeground(java.awt.Color.black); // sets foreground color as black`

- To set Visibility
`levelGauge.setVisible(true); // sets visibility as true`

- To set Enability
`levelGauge.setEnabled(true); // sets enability as true`

- To set Locale
`levelGauge.setLocale(java.util.Locale.ENGLISH); // sets locale as English`

- To set Cursor
`levelGauge.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND_CURSOR)); // sets cursor type as hand cursor`

- To set Bounds
`int x = 5;
int y = 5;
int width = 300;
int height = 200;
levelGauge.setBounds(x, y, width, height); // sets bounds to x, y, width and height`

5.2.3 How to configure caption properties in ElegantJ Level Gauge?

ElegantJ LevelGauge provides following attributes for caption

- Caption text
- Caption font
- Caption color
- Caption location

- To set caption text

```
String caption = "ElegantJ Level Gauge";
levelGauge.setCaption(caption); // sets caption text
```
- To set caption font

```
levelGauge.setFont(new java.awt.Font("Dialog", java.awt.Font.PLAIN,
12)); // sets font as dialog, plain and of size 12
```
- To set caption color

```
java.awt.Color captionColor = java.awt.Color.green;
levelGauge.setCaptionColor(captionColor); // sets caption color as green
```
- To set caption location

```
Point location = new Point(50, 25);
levelGauge.setCaptionLocation(location); // sets caption location
```

5.2.4 How to configure color properties in ElegantJ Level Gauge?

ElegantJ LevelGauge facilitates user to configure following color properties,

- Border color
- Zone color
- Warning zone color
- Alarm zone color

- To set border color

```
java.awt.Color borderColor = java.awt.Color.black;
levelGauge.setBorderColor(borderColor); // sets border color as black
```
- To set active zone color

```
java.awt.Color zoneColor = java.awt.Color.blue;
levelGauge.setActiveZoneColor(zoneColor); // sets zone color as blue
```
- To set warning zone color

```
java.awt.Color warningZoneColor = java.awt.Color.green;
levelGauge.setActiveWarningZoneColor(warningZoneColor); // sets
warning zone color as green
```
- To set alarm zone color

```
java.awt.Color alarmZoneColor = java.awt.Color.red;
levelGauge.setAlarmZoneColor(alarmZoneColor); // sets alarm zone color
as red
```

5.2.5 How to configure gauge values properties in ElegantJ Level Gauge?

ElegantJ LevelGauge allows user to configure following values,

- Indicator value
- Maximum value
- Minimum value
- Warning level value
- Alarm level value

- To set indicator / current value

```
int currentValue = 10;
levelGauge.setValue(currentValue); // sets current value as 10
```
- To set maximum value

```
int maximumValue = 200;
levelGauge.setMaximum(maximumValue); // sets maximum value as 200
```
- To set minimum value

```
int minimumValue = 0;
levelGauge.setMinimum(minimumValue); // sets minimum value as 0
```
- To set warning level value

```
int warningValue = 80;
levelGauge.setWarning(warningValue); // sets warning value as 80
```
- To set alarm level value

```
int alarmValue = 50;
levelGauge.setAlarm(alarmValue); // sets alarm value as 50
```

5.2.6 How to configure header properties in ElegantJ Level Gauge?

ElegantJ LevelGauge provides following header properties,

- Header text
- Header background
- Header foreground
- Header font
- Header spacing (spacing above and below header text)
- Header visibility

- To set header text

```
String headerText = "Header";
levelGauge.setHeaderText(headerText); // sets header text
```
- To set header background

```
java.awt.Color backgroundColor = java.awt.Color.blue;
levelGauge.setHeaderBackground(backgroundColor); // sets header
background as blue
```
- To set header foreground

```
java.awt.Color foregroundColor = java.awt.Color.white;
levelGauge.setHeaderForeground(foregroundColor); // sets header
foreground as white
```
- To set header font

```
levelGauge.setHeaderFont(new java.awt.Font("Dialog",
java.awt.Font.PLAIN, 12)); // sets header font as dialog, plain and of size
12
```
- To set header spacing (spacing above and below header)

```
int headerSpacing = 10;
levelGauge.setHeaderSpacing(headerSpacing); // sets header spacing
(spacing above and below header text) as 10 pixels
```
- To set header visibility

```
levelGauge.setHeaderVisible(true); // sets header visibility as true
```

5.2.7 How to configure footer properties in ElegantJ Level Gauge?

ElegantJ LevelGauge provides following footer properties,

- Footer text
- Footer background
- Footer foreground
- Footer font
- Footer spacing (spacing above and below footer text)
- Footer visibility

- To set footer text

```
String footerText = "Footer";  
levelGauge.setFooterText(footerText); // sets footer text
```

- To set footer background color

```
java.awt.Color backgroundColor = java.awt.Color.blue;  
levelGauge.setFooterBackground(backgroundColor); // sets footer  
background color as blue
```

- To set footer foreground color

```
java.awt.Color foregroundColor = java.awt.Color.white;  
levelGauge.setFooterForeground(foregroundColor); // sets footer  
foreground color as white
```

- To set footer font

```
levelGauge.setFooterFont(new java.awt.Font("Dialog",  
java.awt.Font.PLAIN, 12)); // sets footer font as dialog, plain and of size  
12
```

- To set footer spacing (spacing above and below footer text)

```
int footerSpacing = 10;  
levelGauge.setFooterSpacing(footerSpacing); // sets footer spacing  
(spacing above and below footer text) as 10 pixels
```

- To set footer visibility

```
levelGauge.setFooterVisible(true); // sets footer visibility as true
```

5.2.8 How to configure tick properties in ElegantJ Level Gauge?

ElegantJ LevelGauge provides following tick properties:

- Major tick length
- Minor tick length
- Major tick width
- Minor tick width
- Major tick color
- Minor tick color
- Major tick gap
- Minor tick gap
- Visibility

- To set Major tick length
`int majorTickLength = 7;`
`levelGauge.setMajorTickLength(majorTickLength);` // sets major tick length as 7 pixels
- To set Minor tick length
`int minorTickLength = 5;`
`levelGauge.setMinorTickLength(minorTickLength);` // sets minor tick length as 5 pixels
- To set Major tick width
`int majorTickWidth = 5;`
`levelGauge.setMajorTickWidth(majorTickWidth);` // sets major tick width as 5 pixels
- To set Minor tick width
`int minorTickWidth = 2;`
`levelGauge.setMinorTickWidth(minorTickWidth);` // sets minor tick width as 2 pixels
- To set Major tick color
`levelGauge.setMajorTickColor(java.awt.Color.black);` // sets major tick color as black
- To set Minor tick color
`levelGauge.setMinorTickColor(java.awt.Color.green);` // sets minor tick color as green
- To set Major tick spacing (gap)
`int majorTickSpacing = 5;`
`levelGauge.setMajorTickSpacing(majorTickSpacing);` // sets major tick spacing (gap) as 5 pixels
- To set Minor tick spacing (gap)
`int minorTickSpacing = 5;`
`levelGauge.setMinorTickSpacing(minorTickSpacing);` // sets major tick spacing (gap) as 5 pixels
- To set tick values visibility
`levelGauge.setTickValuesVisible(false);` // sets tick value visibility as false

5.2.9 How to configure mouse activity to move the indication in ElegantJ Level Gauge?

ElegantJ LevelGauge provides feature to set gauge value on mouse-click or mouse-drag.

The possible values for mouse activity are

- `com.elegantj.gauges.level.LevelGauge.MOVE_ON_MOUSE_DRAG` - Value can be changed on mouse-drag
 - `com.elegantj.gauges.level.LevelGauge.MOVE_ON_MOUSE_PRESS` - Value can be changed on mouse-click
 - `com.elegantj.gauges.level.LevelGauge.MOVE_OFF` - User can not interact with gauge on mouse event
- To set mouse activity

```
levelGauge.setActiveIndicatorOn(com.elegantj.gauges.level.LevelGauge.MOVE_ON_MOUSE_DRAG);
```

This allows to set gauge value on mouse drag.

5.2.10 How to configure gauge properties in ElegantJ Level Gauge?

ElegantJ LevelGauge provides following gauge properties,

- Gauge type
- Inner Border
- Outer Border

- To set gauge type
 - Permissible values for gauge type are
 - `com.elegantj.gauges.level.LevelGauge.LEFT_RIGHT_INDICATOR`
 - `com.elegantj.gauges.level.LevelGauge.RIGHT_LEFT_INDICATOR`
 - `com.elegantj.gauges.level.LevelGauge.TOP_BOTTOM_INDICATOR`
 - `com.elegantj.gauges.level.LevelGauge.BOTTOM_TOP_INDICATOR`

```
levelGauge.setIndicatorType(LEFT_RIGHT_INDICATOR);
```

This sets gauge type as left to right

- To set inner border
 - Permissible values for inner border types are
 - `com.elegantj.gauges.GaugeBorder.NONE`
 - `com.elegantj.gauges.GaugeBorder.SIMPLE`
 - `com.elegantj.gauges.GaugeBorder.RAISED`
 - `com.elegantj.gauges.GaugeBorder.LOWERED`

```
int borderType = com.elegantj.gauges.GaugeBorder.SIMPLE;  
int borderWidth = 4;  
java.awt.Color borderColor = java.awt.Color.green;  
com.elegantj.gauges.GaugeBorder gaugeBorder = new  
com.elegantj.gauges.GaugeBorder(borderType, borderWidth,  
borderColor);  
levelGauge.setInnerBorder(gaugeBorder); // sets inner border for gauge
```

- To set outer border

```
int borderType = com.elegantj.gauges.level.GaugeBorder.SIMPLE;  
int borderWidth = 4;  
java.awt.Color borderColor = java.awt.Color.green;  
com.elegantj.gauges.GaugeBorder gaugeBorder = new  
com.elegantj.gauges.GaugeBorder(borderType, borderWidth,  
borderColor);  
levelGauge.setOuterBorder(gaugeBorder); // sets outer border for gauge
```

5.3 ElegantJ Needle Gauge

5.3.1 How to create an instance of ElegantJ NeedleGauge?

To create an instance of ElegantJ NeedleGauge with default constructor,
`com.elegantj.gauges.needle.NeedleGauge needleGauge = new com.elegantj.gauges.needle.NeedleGauge();`

5.3.2 How to apply formatting attributes in ElegantJ Needle Gauge?

ElegantJ Needle Gauge provides following formatting attributes

- Font
- Background
- Foreground
- Visible
- Enabled
- Locale
- Cursor
- Bounds

- To set Font
`needleGauge.setFont(new java.awt.Font("Dialog", java.awt.Font.PLAIN, 12)); // sets font as dialog, style as plain, size as 12`
- To set Background color
`needleGauge.setBackground(java.awt.Color.cyan); // sets background color as cyan`
- To set Foreground color
`needleGauge.setForeground(java.awt.Color.black); // sets foreground color as black`
- To set Visibility
`needleGauge.setVisible(true); // sets visibility as true`
- To set Enability
`needleGauge.setEnabled(true); // sets enability as true`
- To set Locale
`needleGauge.setLocale(java.util.Locale.ENGLISH); // sets locale as English`
- To set Cursor
`needleGauge.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND_CURSOR)); // sets cursor type as hand cursor`
- To set Bounds
`int x = 5;
int y = 5;
int width = 300;
int height = 200;`

```
needleGauge.setBounds(x, y, width, height); // sets bounds to x, y, width and height
```

5.3.3 How to configure caption properties in ElegantJ Needle Gauge?

ElegantJ Needle Gauge provides following customizable caption properties.

- Caption
- Font
- Number's font

- To set caption

```
String caption = "ElegantJ Needle Gauge";  
needleGauge.setCaption(caption); // sets caption
```

- To set caption font

```
needleGauge.setCaptionFont(new java.awt.Font("Dialog",  
java.awt.Font.PLAIN, 12)); // sets caption font as dialog, plain and of size 12
```

- To set number's font

```
needleGauge.setNumbersFont(new java.awt.Font("Dialog",  
java.awt.Font.PLAIN, 12)); // sets number's font as dialog, plain and of size 12
```

5.3.4 How to configure color properties in ElegantJ Needle Gauge?

ElegantJ NeedleGauge allows customizable panel color

- To set panel color

```
java.awt.Color panelColor = java.awt.Color.blue;  
needleGauge.setPanelColor(panelColor); // sets panel color as blue
```

5.3.5 How to configure gauge values properties in ElegantJ Needle Gauge?

ElegantJ NeedleGauge allows user to configure following values properties,

- Maximum value
- Minimum value
- New zone value
- Needle value
- Unit value
- Grad value

- To set maximum value

```
double maximumValue = 320;  
needleGauge.setMaximum(maximumValue); // sets maximum value as 320
```

- To set minimum value

```
double minimumValue = 0;  
needleGauge.setMinimum(minimumValue); // sets minimum value as 0
```

- To set new zone value


```
java.util.Vector zoneValues = new java.util.Vector();
java.util.Vector v1 = new java.util.Vector();

v1.addElement(new com.elegantj.gauges.MeterZone("Safe
Zone",0.0,60.0,Color.cyan,10,0));
v1.addElement(new java.lang.Double(0));
v1.addElement(new java.lang.Double(100));
zoneValues.addElement(v1);

java.util.Vector v2 = new java.util.Vector();

v2.addElement(new com.elegantj.gauges.MeterZone("Warning
Zone",60.0,80.0,Color.blue,5,4));
v2.addElement(new java.lang.Double(0));
v2.addElement(new java.lang.Double(100));
zoneValues.addElement(v2);

java.util.Vector v3 = new java.util.Vector();
v3.addElement(new com.elegantj.gauges.MeterZone("Alarm
Zone",80.0,100.0,Color.red,5,4));
v3.addElement(new java.lang.Double(0));
v3.addElement(new java.lang.Double(100));
zoneValues.addElement(v3);

needleGauge.setNewZoneValues(zoneValues);
```
- To set needle value


```
double needleValue = 13;
needleGauge.setNeedleValue(needleValue); // sets needle value as 13
```
- To set unit value


```
String unitString = "Km/hr";
needleGauge.setUnit(unitString); // sets unit value
```
- To set gradient of scale of gauge


```
double gradValue = 50;
needleGauge.setScaleGrad(gradValue); // sets gradient of scale of gauge
```

5.4 ElegantJ Thermometer Gauge

5.4.1 How to create an instance of ElegantJ ThermometerGauge?

To create an instance of ElegantJ ThermometerGauge default constructor,

```
com.elegantj.gauges.thermo.ThermometerGauge thermometerGauge = new
com.elegantj.gauges.thermo.ThermometerGauge();
```

5.4.2 How to apply formatting attributes in ElegantJ Thermometer Gauge?

ElegantJ ThermometerGauge provides following formatting and appearance attributes -

- Font
- Background
- Foreground
- Visible
- Enabled
- Locale
- Cursor
- Bounds

- To set Font


```
thermometerGauge.setFont(new java.awt.Font("Dialog",
java.awt.Font.PLAIN, 12)); // sets font as dialog, plain and of size 12
```

- To set Background color


```
thermometerGauge.setBackground(java.awt.Color.cyan); // sets
background color as cyan
```

- To set Foreground color


```
thermometerGauge.setForeground(java.awt.Color.black); // sets
foreground color as black
```

- To set Visibility


```
thermometerGauge.setVisible(true); // sets visibility as true
```

- To set Enability


```
thermometerGauge.setEnabled(true); // sets enability as true
```

- To set Locale


```
thermometerGauge.setLocale(java.util.Locale.ENGLISH); // sets locale as
English
```

- To set Cursor


```
thermometerGauge.setCursor(new
java.awt.Cursor(java.awt.Cursor.HAND_CURSOR)); // sets cursor type as
hand cursor
```

- To set Bounds


```
int x = 5;
int y = 5;
int width = 300;
int height = 200;
thermometerGauge.setBounds(x, y, width, height); // sets bounds to x, y,
width and height
```

5.4.3 How to configure caption properties in ElegantJ Thermometer Gauge?

ElegantJ Thermometer gauge provides following caption properties

- Caption
- Caption font
- Number's font

- To set caption


```
String caption = "Caption";
thermometerGauge.setCaption(caption); // sets caption
```

- To set caption font
`thermometerGauge.setCaptionFont(new java.awt.Font("Dialog", java.awt.Font.PLAIN, 12)); // sets caption font as dialog, plain and of size 12`
- To set number's font
`thermometerGauge.setNumbersFont(new java.awt.Font("Dialog", java.awt.Font.PLAIN, 12)); // sets number's font as dialog, plain and of size 12`

5.4.4 How to configure color properties in ElegantJ Thermometer Gauge?

ElegantJ ThermometerGauge provides the following color properties.

- Panel color
- Bulb color
- To set panel color
`java.awt.Color panelColor = java.awt.Color.blue;`
`thermometerGauge.setPanelColor(panelColor); // sets panel color as blue`
- To set bulb color
`java.awt.Color bulbColor = java.awt.Color.yellow;`
`thermometerGauge.setBulbColor(bulbColor); // sets bulb color as yellow`

5.4.5 How to configure gauge values properties in ElegantJ Thermometer Gauge?

You can set following values for ElegantJ ThermometerGauge:

- Indicator/current value
- Indicator width
- Maximum value
- Minimum value
- Minor tick value
- Scaling value of gradient
- To set indicator (current) value
`int currentValue = 50;`
`thermometerGauge.setValue(currentValue); // sets indicator (current) value`
- To set width of indicator value
`int widthValue = 10;`
`thermometerGauge.setIndicatorWidth(widthValue); // sets width of indicator as 10 pixels`
- To set maximum value
`int maximumValue = 200;`
`thermometerGauge.setMaximum(maximumValue); // sets maximum value as 200`
- To set minimum value
`int minimumValue = 0;`

```
thermometerGauge.setMinimum(minimumValue); // sets minimum value  
as 0
```

- To set Major tick spacing value

```
double majorTickValue = 10;  
thermometerGauge.setMajorTick(minorTickValue); // sets major tick  
spacing value as 10
```
- To set Minor tick spacing value

```
double minorTickValue = 5;  
thermometerGauge.setMinorTick(minorTickValue); // sets minor tick  
spacing value as 5
```

6 Product and Support Information

Product & Support Information:

- You can find more information about ElegantJ Charts Designer and its features on www.ElegantJCharts.com.
- If you looking for further support apart from this documentation, then you can purchase our support package from the website.
- Forward your sales related mail to sales@ElegantJCharts.com.

Feedback and Suggestions:

- We will be pleased to get your feedback as well as suggestions about our products.
- Forward any feedback or suggestions related mails to support@ElegantJCharts.com.