GAUGES

Description/Introduction

Gauges are an important part of the LOOPS user interface for both developers and end users. Gauges assist in understanding the dynamic nature of the programs. This is in contrast to the more typical case of debugging programs using static means. In the creation of user-friendly interfaces, you can use gauges to display, in analog or digital form, various data that may be changing. Also, by employing active gauges, you can provide a convenient way to interact with a system.

One of the features of gauges is the ease with which you can use them in a system. In more traditional languages, if you want to understand how a variable is changing over the course of a computation, you must make modifications in your program wherever you want to begin or end the examination of a variable. Given the capabilities of active values used by gauges, you need only attach or detach a gauge to the data that you are interested in monitoring.

The following types of gauges are available:

- Meter; a circular instrument that wraps around any number of times.
- Dial; a bounded dial, like an automobile speedometer.
- LCD; a gauge that uses the entire window to display a value.
- Scale; a horizontal or vertical display of a gauge.
- ActiveScale; a scale that allows you to change the gauge value.

Gauges are an example of the combination of programming capabilities within LOOPS. The different types of gauges are defined within the context of an inheritance lattice. This allows the more general functionality and variables to be allocated to more general gauge classes, with specific functionality placed in more restricted classes. You can also see the use of mixins to add a small amount of functionality to several different classes of gauges.

Note: Mixins are classes that are used only in conjunction with another class to create a subclass.

The methods within gauges are built upon both function calling and message sending. Gauges are "attached" to objects through the mechanism of active values. Since gauges are built upon the mechanism of active values, gauges can only be attached to data within objects. It is not possible to use gauges to monitor any arbitrary Lisp variable.

Prerequisites

The default font for gauges is Modern 10.

Installation/Loading Instructions

Gauges are divided among several different files to allow you to load only those objects and functions that you need. The table below lists the files to load for each type of gauge. The filecoms for each file will try to load any other required gauge files from **LOOPSLIBRARYDIRECTORY**. The file GAUGES.DFASL and either GAUGEINSTRUMENTS.DFASL or GAUGEALPHANUMERICS.DFASL will always be loaded; other files may also be loaded.

Gauge	File to load
LCD	GAUGEALPHANUMERICS.DFASL
METER	GAUGEMETERS.DFASL
DIAL	GAUGEDIALS.DFASL
SCALE	GAUGECALES.DFASL
ACTIVE SCALE	GAUGEACTIVE.DFASL

Additionally, the file GAUGESELFSCALEMIXIN.DFASL can be loaded to add the class **SelfScaleMixin**, and GAUGEALARMS.DFASL can be loaded to add the class **AlarmMixin**.

To load the required files, first set the value of **LOOPSDIRECTORY** to include the directory where the gauges files are stored, then type the following expression in the Executive:

(LOAD '*FILENAME*)

To load all of the gauges, load the file GAUGELOADER and then enter `(LOADGAUGES)`. GAUGELOADER also sets the variables: **GAUGEFILES** and **GaugeClasses**.

(**LOADGAUGES** *LDFLG SOURCES?FLG*) [Function]

Purpose: Loads all the gauges.

Behavior: Assumes that all of the gauge files are on the **LOOPSDIRECTORY** search path.

All the gauge files will be loaded based upon the settings of *LDFLG* and *SOURCES?FLG*. A **FILESLOAD** expression is built up and evaluated.

Arguments:

LDFLG Can be NIL, PROP, or SYSLOAD. See the **LDFLG** discussion under loading in the *Interlisp-D Reference Manual*.

SOURCES?FLG
Can be NIL or T. If NIL, this attempts to load the compiled files before trying to load the sources. If T, only the sources are loaded.

Returns: Used for side effect only.

GAUGEFILES [Variable]

Behavior: Initialized to (GAUGEACTIVE GAUGEALARMS GAUGEALPHANUMERICS GAUGEBOUNDEDMIXIN GAUGEDIALS GAUGEDIGIMETER GAUGEDIGISCALE GAUGEINSTRUMENTS GAUGEMETERS GAUGES GAUGESCALES GAUGESELFSCALEMIXIN)

GaugeClasses [Variable]

Behavior: Initialized to (GaugeAV ActiveGaugeMixin Gauge AlarmMixin BoundedMixin SelfScaleMixin)

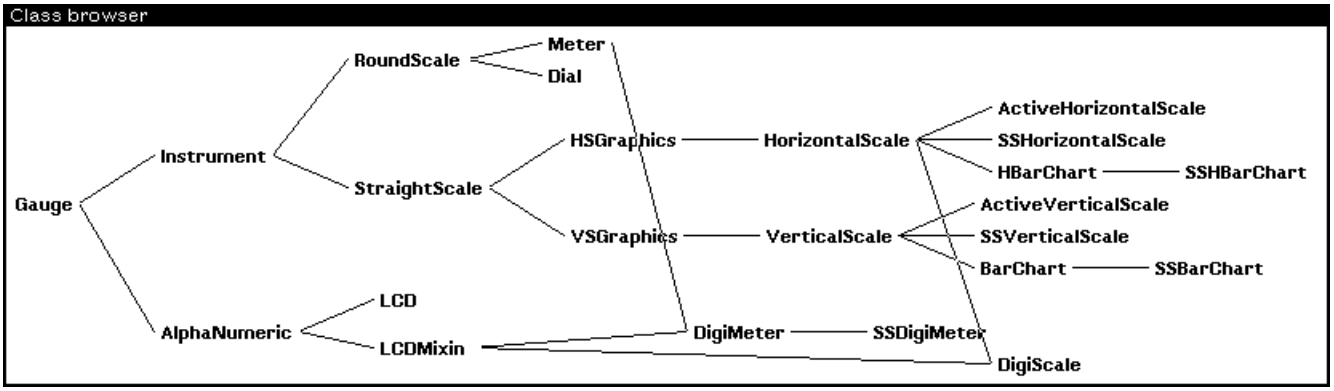
Call (Browse **GaugeClasses**) to open a browser of all of the gauge classes.

Application/Module Functionality

This section describes the gauge classes and methods.

Gauge Classes

This section describes the available gauges shown in the following browser.



Note: The browser does not include the optional mixin classes.

Within the class description of each class, the instance variables and class variables that are specializations only because they have different default values are not listed.

Name	Type	Description
ActiveGaugeMixin	AbstractClass	A gauge class that allows you to set the value of the variable being monitored with the cursor, via a SET menu.
ActiveHorizontalScale	Class	An active gauge that displays the value on a horizontal scale.
ActiveVerticalScale	Class	An active gauge that displays its value on a vertical scale.
AlarmMixin	AbstractClass	A mixin that adds alarm functionality to any gauge.
AlphaNumeric	AbstractClass	A gauge that gives an alphanumeric display of a value.

BarChart	Class	A gauge that displays more than one **VerticalScale** side-by side.
BoundedMixin	AbstractClass	Creates a bounded scale for **displayVal**; to be used as a mixin for instruments.
Dial	Class	A bounded dial, like an automobile speedometer.
DigiMeter	Class	A gauge that displays both an **LCD** and a meter.
DigiScale	Class	A gauge that displays both an **LCD** and a horizontal scale.
Gauge	AbstractClass	A class for objects that present a dynamic graphical image of a LOOPS value.
GaugeAV	Class	An active value associated with a gauge.
HBarChart	Class	A gauge that displays more than one **HorizontalScale** side-by side.
HorizontalScale	Class	A labeled, bounded scale with a bar that fills to the right.
HSGraphics	AbstractClass	Gauge that is displayed in the form of a singe horizontal scale or bar.
Instrument	AbstractClass	A numeric gauge that is externally scaled by **inputLower** and **inputRange** and scaled internally by **lower** and **range**.
LCD	Class	Differs from **AlphaNumeric** in that the entire gauge window is the printing region.
LCDMixin	AbstractClass	Computes print region differently from **LCD**.
Meter	Class	A circular instrument that wraps around any number of times.
RoundScale	AbstractClass	Abstract Class for instruments with circular (arc) scales.
SelfScaleMixin	AbstractClass	Provides for the gauge to rescale according to the reading.
SSBarChart	Class	A self-scaling version of **BarChart.**
SSDigiMeter	Class	A self-scaling version of **DigiMeter**.
SSHBarChart	Class	A self-scaling version of **HBarChart.**
SSHorizontalScale	Class	Gauge that is displayed in the form of a single scale or bar which rescales itself accordingly.
SSVerticalScale	Class	Gauge that is displayed in the form of a single vertical scale or bar which rescales itself accordingly.
StraightScale	AbstractClass	Abstract Class for instruments with straight scales.
VSGraphics	AbstractClass	Gauge that is displayed in the form of a single vertical scale or bar.
VerticalScale	Class	Gauge that is displayed in the form of a single vertical scale or bar.

ActiveGaugeMixin [Class]

Description: A gauge class that allows you to set the value of the variable being monitored with the cursor, via a **SET** menu.

MetaClass: AbstractClass

Supers: Object

Class Variables: None.

Instance Variables: **cursor** The cursor to use when changing the scale; the default is NIL.

ActiveHorizontalScale [Class]

Description: An active gauge that displays the value on a horizontal scale. This gauge shows the value of the data it is connected with and allows you to change that data with the gauge.

MetaClass: Class

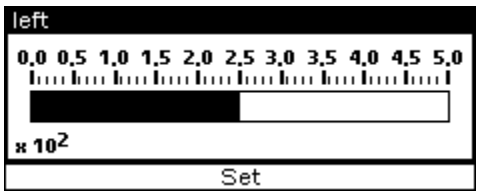
Supers: ActiveGaugeMixin, HorizontalScale

Class Variables: None.

Instance Variables: **cursor** Cursor to use when changing the scale; its property **:initform** is set to **HorizontalAGCursor**.

Example: These gauges have an attached menu at the bottom of the gauge. When you position the cursor over this menu and press a mouse button, the cursor changes to the following shape:

While the left button is held down, the system tracks movements of the cursor and changes the value that the gauge is monitoring.



ActiveVerticalScale [Class]

Description: Similar to **ActiveHorizontalScale**, except that a vertical scale is used.

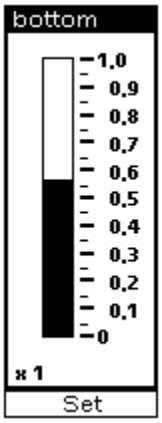
MetaClass: Class

Supers: ActiveGaugeMixin, VerticalScale

Class Variables: None.

Instance Variables: **cursor** Cursor to use when changing the scale; its **:initform** property is set to **VerticalAGCursor**.

Example: Similar to **ActiveHorizontalScale**. When setting, the cursor changes to the following shape:



AlarmMixin [Class]

Description: A mixin that adds alarm functionality to any gauge. An alarm is defined as warning object that is set off when the value being monitored falls outside of the specified range. The gauge flashes and stays inverted when the alarm is tripped.

CAUTION

When a new class of gauges is created that will use the properties of **AlarmMixin**, **AlarmMixin** should be the first class on the Supers list of the new class. This guarantees that the **AlarmMixin.Set** method is invoked.

MetaClass: AbstractClass

Supers: Object

Class Variables: MiddleButtonItems

Instance Variables:

lowTripPoint
Alarm is triggered when reading goes below this point.

hiTripPoint Alarm is triggered when reading goes above this point.

flashNumber
Number of times alarm will flash when it is tripped.

flashInverval
Interval in milliseconds between flashes.

AlphaNumeric [Class]

Description: This class contains some of the methods and data for the LCD classes. These gauges can display any type of character, letters, or numbers.

MetaClass: AbstractClass

Supers: Gauge

Class Variables: None.

Instance Variables: **precision** Number of characters displayed in the reading. The default value is 5.

BarChart [Class]

Description: A gauge that can display more than one **VerticalScale** at once, side-by side.

MetaClass: Class

Supers: VerticalScale

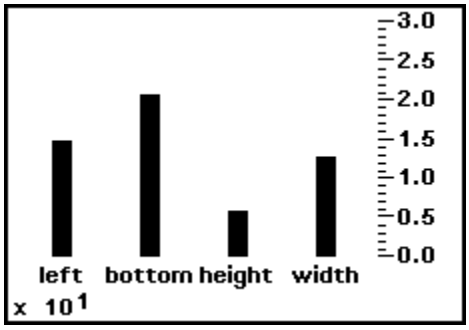
Class Variables: None.

Instance Variables: **maxLabelWidth**
Maximum width of labels on each bar. Default value is 0 which means no limit.

scaleLeft
Offset within the gauge window from the left for the leftmost bar. Default value is 3.

scaleBottom
Offset within the gauge window from the bottom for all the bars. Default value is 30.

Example: Here is a **BarChart** showing the size and shape of a window. It is displaying the values 15, 21, 13, and 6.



BoundedMixin [Class]

Description: This mixin is a super of the scale classes and **Dial**. If a gauge that has **BoundedMixin** as a super class tries to display a new setting that is outside of the range of the gauge, the gauge will display the minimum or maximum value as appropriate and place a "??" in the window.

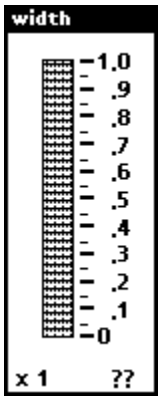
MetaClass: AbstractClass

Supers: Object

Class Variables: None.

Instance Variables: None.

Example: Here is a vertical scale that displays a reading greater than its maximum.



Dial [Class]

Description: A bounded dial, like an automobile speedometer.

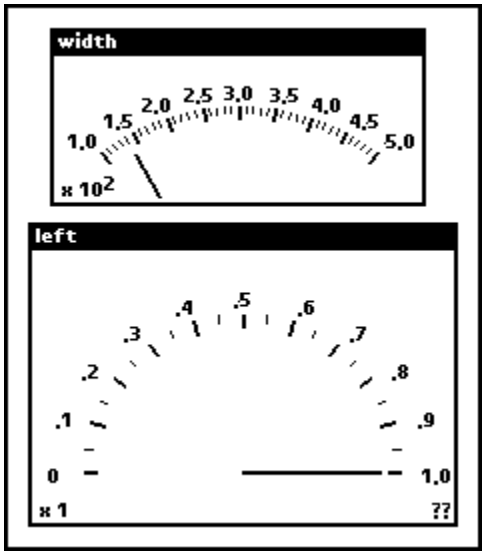
MetaClass: Class

Supers: BoundedMixin, RoundScale

Class Variables: None.

Instance Variables: This class specializes the same instance variables as **RoundScale**.

Example: The angle of the arc changes with the shape of the window.



DigiMeter [Class]

Description: A gauge that combines both a meter and an LCD.

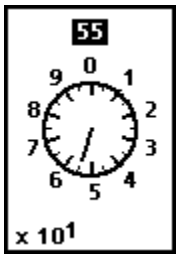
MetaClass: Class

Supers: Meter, LCDMixin

Class Variables: None.

Instance Variables: **spaceForLCD**
Vertical space required by LCD within the gauge. Defaults to 30.

Example: This **DigiMeter** is displaying 55.



DigiScale [Class]

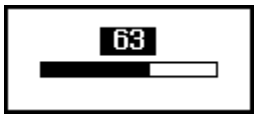
Description: A gauge that combines both a horizontal scale having no ticks and an LCD.

MetaClass: Class

Supers: HorizontalScale, LCDMixin

Class Variables: None.

Example: This **DigiScale** is displaying 63 with its scale set from 0 to 100.



Gauge [Class]

Description: A class for objects that present a dynamic graphical image of a LOOPS value. This class provides most of the methods for using gauges.

MetaClass: AbstractClass

Supers: Window

Class Variables:

LeftButtonItems
Menu options associated with the left mouse button.

MiddleButtonItems
Menu options associated with the middle mouse button.

Instance Variables:

reading External value of reading. The default value is 0.

containedInAV
Active value that connects the gauge to the data it is monitoring. It should be an instance of the class **GaugeAV**.

font Font that is used by a gauge; default value is (Modern 10).

width Width of a gauge; has property **min,** which specifies the minimum width for a gauge.

height Height of a gauge; has property **min**, which specifies the minimum height for a gauge.

GaugeAV [Class]

Description: An active value that is associated with a gauge.

MetaClass: Class

Supers: LocalStateActiveValue

Class Variables: None.

Instance Variables:

gauge The gauge connected to this active value.

object The object containing the variable associated with the active value.

propName The property name of the associated variable.

type Data type of the associated variable.

varName Name of the associated variable.

HBarChart [Class]

Description: A gauge that can display more than one **HorizontalScale** at once, side-by side.

MetaClass: Class

Supers: HorizontalScale

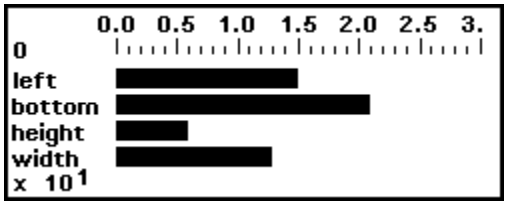
Class Variables: None.

Instance Variables:

maxLabelWidth
Maximum width of labels on each bar. Default value is 0 which means no limit.

scaleLeft
Offset within the gauge window from the left for the leftmost bar. Default value is 3.

Example: Here is an **HBarChart** showing the size and shape of a window. It is displaying the values 15, 21, 13, and 6.



HorizontalScale [Class]

Description: A labeled, bounded scale with a bar that fills to the right.

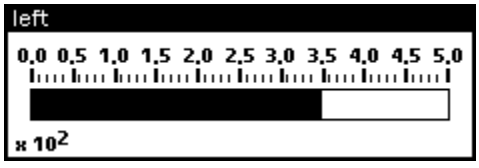
MetaClass: Class

Supers: HSGraphics

Class Variables: None.

Instance Variables: None.

Example: This **HorizontalScale** is reading 350 on a scale from 0 to 500.



HSGraphics [Class]

Description: This class provides some of the methods for displaying the graphics of a horizontal scale.

MetaClass: AbstractClass

Supers: StraightScale

Class Variables: None.

Instance Variables:

scaleBottom Bottom edge of scale in pixels. The default value is 10.

scaleLeft Left edge of scale in pixels. The default value is 12.

scaleWidth Width of inside of scale in pixels. The default value is 120.

scaleHeight Height of scale in pixels. The default value is 15.

Instrument [Class]

Description: A class that provides additional methods and data for gauges that display only numerical data. This data is externally scaled by **inputLower** and **inputRange**, and scaled internally by **lower** and **range**.

MetaClass: AbstractClass

Supers: Gauge

Class Variables: None.

Instance Variables:

ticks Scale marks on the instrument; value is a number or NIL; **smallTicks** property indicates the number of smaller ticks between each large tick.

displayVal Internal value relative to instrument.

range Range for internal **displayVal** .

inputRange Range for external reading.

lower Lower bound for internal **displayVal**.

inputLower Lower bound for external reading.

brushWidth Scale factor for width of ticks, rays, and circles in pixels.

labels The labels that will be displayed on the gauge.

labelScale A dotted pair representing the sign and exponent of a reading.

spaceForLabelScale Extra vertical space to display scale label.

LCD [Class]

Description: Differs from **LCDMixin** in that the entire gauge window is the printing region.

MetaClass: Class

Supers: AlphaNumeric

Class Variables: None.

Instance Variables: None.

Example: This **LCD** is displaying the string "Mumble", and has been **Shape**dto 120 x 60.

LCDMixin [Class]

Description: Computes printing region differently from LCD so that an LCD may be added into another window.

MetaClass: AbstractClass

Supers: AlphaNumeric

Class Variables: None.

Instance Variables:

precision Number of characters displayed in the reading; the default value is 3. Its property is **readingRegion**; the default value is NIL.

readingY Y position of bottom of reading. The default value is 7.

Meter [Class]

Description: A circular instrument that wraps around any number of times. It displays a sign and exponent in the lower left corner of its window.

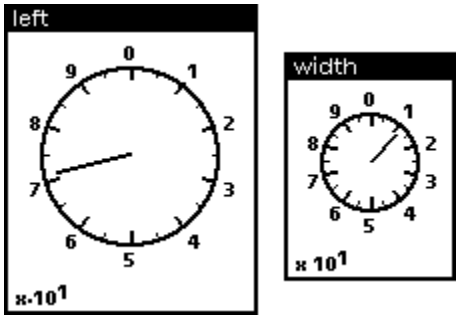
MetaClass: Class

Supers: RoundScale

Class Variables: None.

Instance Variables: This class specializes the same instance variables as **RoundScale**.

Example: The **Meter** on the left is displaying a negative value.



RoundScale [Class]

Description: Abstract Class for instrumentswith circular (arc) scales.

MetaClass: AbstractClass

Supers: Instrument

Class Variables: None.

Instance Variables:

needleLength
Radius of needle in pixels. The default value is 15.

radius Radius of arc in pixels. The default value is 10.

xc x-coordinate window coordinate of center of arc. (See **DRAWARC** in the *Lisp Release Notes*.)

yc y-coordinate window coordinate of center of arc. (See **DRAWARC** in the *Lisp Release Notes*.)

SelfScaleMixin [Class]

Description: Provides for the gauge to rescale according to the reading.

MetaClass: AbstractClass

Supers: Object

Class Variables: None.

Instance Variables: **lowScaleFactor**
Rescales if reading shrinks so that it will fit more than **lowScaleFactor** times in **inputRange**. The default value is 5.

SSBarChart [Class]

Description: A self-scaling version of **BarChart.**

MetaClass: Class

Supers: BarChart

Class Variables: None.

Instance Variables: None.

SSDigiMeter [Class]

Description: A self-scaling version of **DigiMeter**.

MetaClass: Class

Supers: DigiMeter

Class Variables: None.

Instance Variables: None.

SSHBarChart [Class]

Description: A self-scaling version of **HBarChart.**

MetaClass: Class

Supers: HBarChart

Class Variables: None.

Instance Variables: None.

SSHorizontalScale [Class]

Description: Gauge that is displayed in the form of a single horizontal scale or bar which rescales itself accordingly.

MetaClass: Class

Supers: VerticalScale

Class Variables: None.

Instance Variables: None.

SSVerticalScale [Class]

Description: Gauge that is displayed in the form of a single vertical scale or bar which rescales itself accordingly.

MetaClass: Class

Supers: HorizontalScale

Class Variables: None.

Instance Variables: None.

StraightScale [Class]

Description: Abstract class for instruments with straight scales.

MetaClass: AbstractClass

Supers: BoundedMixin, Instrument

Class Variables: None.

Instance Variables: **shade** Shade of bar; numeric value from 0 to 65535. The default value is 65535, which is BLACKSHADE.

VerticalScale [Class]

Description: Gauge that is displayed in the form of a singe vertical scale or bar.

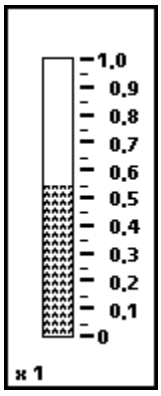
MetaClass: Class

Supers: VSGraphics

Class Variables: None.

Instance Variables: None.

Example: This **VerticalScale** is displaying the value .55 and has its **Shade** set to 1258.



VSGraphics [Class]

Description: Similar to **HSGraphics** but for vertical scales.

MetaClass: AbstractClass

Supers: StraightScale

Class Variables: None.

Instance Variables:

scaleBottom Bottom edge of scale in pixels. The default value is 12.

scaleLeft Left edge of scale in pixels. The default value is 15.

scaleWidth Width of inside of scale in pixels. The default value is 15.

scaleHeight Height of scale in pixels. The default value is120.

Gauge Methods

This section describes the available methods and functions which are used to manipulate gauges. In many cases, a particular gauge class specializes a method defined in the class **Gauge**. In this case, the specialized method definition is not explicitly defined; instead, this is noted in the Specializes/Specializations field of the description.

Name	Type	Description
Attach	Method	Connects a gauge to an object**.**
Attached?	Method	Determines what the gauge is attached to.
ChangeFont	Method	Sets the gauge's instance variable **font** and updates the gauge.
Close	Method	Detaches the gauge and closes the window.
Destroy	Method	Destroys the gauge, detaching it first.
Detach	Method	Detaches the gauge from the variable it is attached to.
Reset	Method	Resets the gauge's instance variable **reading** .
SetScale	Method	Sets the scale for the gauge.
Shape	Method	Sweeps a new region.
ShapeToHold	Method	Shapes the gauge window to its smallest possible size.
Update	Method	Reinitializes the gauge and its display window to reflect the current state.

(← *self* **Attach** *obj varName propName type xOrPos y*) [Method of Gauge]

Purpose: Connects a gauge to an object.

Behavior: Displays the gauge on the screen and associates that gauge with the variable *varName* of *obj*. If *propName* is specified, the gauge will montior the variable's property. If *xOrPos* and *y* are not specified, a small box will appear which must be positioned to place the gauge.

Arguments: *obj* A pointer to the object to which the gauge is to be attached.

varName The name of the instance variable, class variable, or method to which the gauge is to be attached.

propName If non-NIL, the gauge will be attached to this property.

type One of IV, CV, or METHOD, within the object being connected to the gauge. If NIL, it defaults to IV.

xOrPos A numerical value to specify where, in screen coordinates, the gauge will be placed on the display. If NIL, you are asked to place the gauge on the screen. This can be a number to specify the x coordinate or a position. If it is a number, also specify *y* .

y If *xOrPos* is not a position, this specifies the y coordinate in screen coordinates for the gauge.

Returns: *self*

Specializations: **StraightScale.Attach** has an additional *shade* argument so that the shade of the scale may be specified at the time the gauge is attached. The following shows the argument list for this method:

(← ($ *instance OfHorizontalScale*) **Attach** *obj varName shade propName type xOrPos y*)

The **Attach** methods for **BarChart**, **HBarChart**, and their subclasses take an additional *label* argument. If no *label* argument is given, the bar is labeled with *varName*. The *label* argument comes last, as follows:

(← ($ *instance OfBarChart*) **Attach** *obj varName propName propName type xOrPos y label*)

(← *self* **Attached?** *don'tPrintFlg*) [Method of Gauge]

Purpose: Determines what a gauge is attached to.

Behavior: If *don'tPrintFlg* is non-NIL this returns the value of the gauge instance variable **containedInAV**. If *dontPrintFlg* is NIL, the **object** and the **varName** the gauge is attached to will be printed in an attached window.

Arguments: *don'tPrintFlg*
*S*uppresses displaying what the gauge is attached to.

Returns: NIL

(← *self* **ChangeFont** *newFont*) [Method of Gauge]

Purpose/Behavior: Sets the gauge's instance variable **font** to *newFont* and updates the gauge. If the gauge is too small for *newFont*, it is reshaped.

Arguments: *newFont* A font in which to display the gauge's text.

Returns: Previous value of **font**.

(← *self* **Close**) [Method of Gauge]

Purpose/Behavior: Detaches the gauge and closes the window.

Returns: CLOSED

(← *self* **Destroy**) [Method of Gauge]

Purpose/Behavior: Destroys the gauge, detaching it first before closing the window.

Returns: NIL

(← *self* **Detach**) [Method of Gauge]

Purpose/Behavior: Detaches the gauge from the variable to which it is attached. This prints in an attached window that the gauge is being detached, and deletes all of the links connecting the gauge, active value, and object being monitored. Does not close the window.

Returns: NIL

(← *self* **Reset** *newReading*) [Method of Gauge]

Purpose/Behavior: Sets the gauge's instance variable **reading** to *newReading* and updates the gauge. If the gauge is too small for *newReading* and it is SelfScaling, it is reshaped.

Arguments: *newReading*
Sets the instance variable **reading** to *newReading*, and updates the gauge without going through any intermediate steps.

Returns: NIL if gauge is **AlphaNumeric** or **RoundScale**; otherwise *self*.

Specializations: **Alphanumeric.Reset**, **RoundScale.Reset**

Example: The following example causes the LCD to be redisplayed with the *newReading*:

```
13←(← ($ lcd1)  Reset   "New Title")
```

(← *self* **SetScale** *min max*) [Method of Gauge]

Purpose/Behavior: Sets the scale for the gauge; computes the new scale values and redisplays if necessary.

Arguments: *min* Lowest value on scale.

max Highest value on scale.

Returns: *self*

(← *self* **Shape** *newRegion noUpdateFlg*) [Method of Gauge]

Purpose/Behavior: If *newRegion* is NIL, you are prompted to sweep out a region which has a minimum sized based upon a **min** property of **IV width** and **height:,min**. If *newRegion* is non-NIL, it is first checked to guarantee that it is at least as large as **width:,min** by **height:,min**.

Arguments: *newregion* List specifying the external coordinates of the window in which the gauge is displayed; list is of the form (left, bottom, width, height).

noUpdateFlg
If NIL, reshapes the gauge.

Returns: NIL

Specializes: Window

Specializations: **LCD**, **Meter**, **DigiMeter**. **Meter.Shape** has an extra argument *ExtraSpaceFlg*. If T, this will allow you to shape a fairly arbitrary region for the gauge; if NIL, the meter is constrained to be close to a square. This latter behavior is what the user sees when trying to shape the meter from the window menu.

BarChart, HBarChart , and their subclasses can only be freely **Shape**d in the direction their bars run (i.e., **BarChart**s can be **Shape**d vertically and **HBarChart**s can be **Shape**d horizontally). Their size along the other dimension is fixed by the number of values attached to the chart .

Example: This example reshapes the gauge to a location where the lower left corner is at (10,100) a width of 50 and a height of 150.

```
14←(← ($ lcd1) Shape '(10 100 50 150))
```

(← *self* **ShapeToHold**) [Method of Gauge]

Purpose/Behavior: Shapes the gauge window to its smallest possible size based on **width:,min** and **height:,min** and redisplays the gauge.

Returns: NIL

Specializations: **LCD.Shape**

(← *self* **Update**) [Method of Gauge]

Purpose/Behavior: Reinitializes the gauge and its display window to reflect the current state.

Returns: *self*

Categories: Window

Examples

The typical use pattern for a gauge is to first create it, set the scale to the appropriate value, and attach it to the desired data.

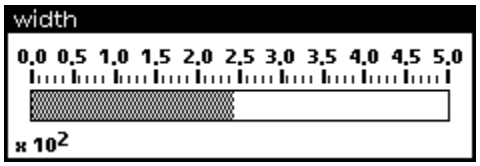
To attach a horizontal scale to a LOOPS window, **w1**, first enter

```
15←(← ($ Window) New 'w1)
#,($& HorizontalScale (|OZW0.1Y:.;h.Qm:| . 495))


16←(← ($ HorizontalScale) New 'hs1)
#,($& HorizontalScale (|OZW0.1Y:.;h.Qm:| . 496))


17←(← ($ hs1) SetScale 0 500)
NIL
```

Now make the connection.

```
18←(← ($ hs1) Attach ($ w1) 'width GRAYSHADE)
#,($& HorizontalScale (|OZW0.1Y:.;h.Qm:| . 496))
```

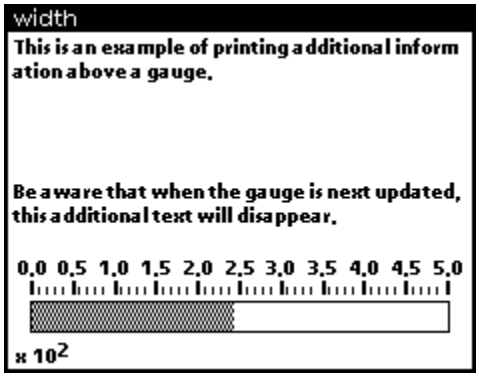
The following gauge appears and you are prompted to place it .



The title of the gauge shows the instance variable being monitored.

Gauges can be shaped larger. The graphics used to display scales do not change; extra white space is added to the top or right. You can use this space to print additional information, as follows:

```
19←(MOVETOUPPERLEFT (@ ($ hs1) window))
{WINDOW}#372,7104

20←(PRIN1 "This is an example of printing additional
information above a gauge.


Be aware that when the gauge is next updated, this
additional text will disappear." (@ ($ hs1) window))
"This is an example of printing additional information
above a gauge.


Be aware that when the gauge is next updated, this
additional text will disappear."
```



Limitations

When a font is changed, a gauge occasionally needs to be updated to be correctly displayed.

Instruments can have only floating point numbers for labels, and cannot have integers.

[This page intentionally left blank]