If *flags* is 2, CmpStr does a binary comparison and returns the following values:

0:	*str1* and *str2* are equal
1:	*str1* and *str2* are not equal

Usually strings do not contain null bytes. Text comparison treats a null byte as meaning "end of string". Binary comparison does not treat a null byte specially. The example below illustrates the treatment of null bytes.

Example

```
Function TestCmpStr()
    String str1 = "A" + num2char(0) + "B"       // num2char(0) is a null byte
    Print strlen(str1)                          // Prints 3

    String str2 = "A" + num2char(0) + "C"
    Print strlen(str2)// Prints 3

    Print CmpStr(str1,str2,0)                   // Prints 0 meaning "equal"
    Print CmpStr(str1,str2,1)                   // Prints 0 meaning "equal"
    Print CmpStr(str1,str2,2)                   // Prints 1 meaning "unequal"
End
```

The "Print strlen" statements illustrate that strlen does not treat null as meaning end of string.

The results from CmpStr with *flags*=0 and *flags*=1 illustrate that CmpStr treated null as meaning "end of string" and consequently compared only the first byte of *str1* to the first byte of *str2*. Both are "A" so CmpStr returned 0 meaning "equal".

The result from CmpStr with *flags*=2 illustrates that CmpStr did not treat null as meaning "end of string" and consequently compared all three bytes of *str1* to all three bytes of *str2*. The third bytes do not match so CmpStr returned 1 meaning "unequal".

See Also

ReplaceString, **Using Strings** on page IV-172

ColorScale

ColorScale [***flags***] [**, *keyword* = *value*,** …] [***axisLabelStr***]

The ColorScale operation adds a color scale (or "color legend") annotation to the a graph, Gizmo plot or page layout. For background information, see **Color Scales** on page III-47.

The ability to add colorscales to a Gizmo plot was added in Igor Pro 8.00.

The ColorScale operation can be executed with no flags and no parameters. It acts on the current target window or the active subwindow or on the window or subwindow specified by the /W flag.

When the target is a graph, the color scale represents the colors and values associated with the first image plot that was added to the graph. If there is no image plot in the graph, the color scale represents the first contour plot or first f(z) trace added to the graph, one of which must exist for the command to execute without error when executed without parameters.

When the target is a Gizmo plot, the color scale represents the colors and values associated with the first surface object, preferring the surface fill color table over the surface gridlines color table. If there is no such surface object, the color scale represents the first scatter object using a color table. If there is no such scatter object, the color scale represents the first path object using a color table. If there is no such path object, the color scale represents the first ribbon object using a color table.

Executing ColorScale with no parameters when the target is a page layout displays a color bar as if the `ctab={0,100,Rainbow}` parameters had been specified.

Flags

Use the /W=*winName* flag to specify a specific graph or layout window. When used on the command line or in a Macro or Proc, /W must precede all other flags.

To change a color scale, use the /C/N=*name* flags. Annotations are automatically named "text0", "text1", etc. if no name is specified when it is created, and you must use that name to modify an existing annotation or else a new one will be created.

For explanations of all flags see the **TextBox** operation.

Parameters

The following keyword-value pairs are the important parameters to the ColorScale operation, because they specify what object the color scale is representing.

For use with page layouts, the keyword ={*graphName*,…} form is required.

For graphs it is simpler to use the image=*imageInstanceName* form (omitting graphName), though you can use $"" to mean the top graph, or specify the name of another graph with the long form. See the **Image Plot ColorScale Examples**.

axisLabelStr Contains the text printed beside the color scale's main axis.

Using escape codes you can change the font, size, style and color of text, create superscripts and subscripts, create dynamically-updated text, insert legend symbols, and apply other effects. See **Annotation Escape Codes** on page III-53 for details.

You can use **AppendText** or **ReplaceText** to modify this axis label string. The default value for axisLabelStr is "".

cindex=*cindexMatrixWave*

The colors shown are those in the named color index wave with axis values derived from the wave's X (row) scaling and units.

The image colors are determined by doing a lookup in the specified matrix wave. See the ModifyImage cindex keyword.

contour=*contourInstanceName*

contour={*graphName*,*contourInstanceName*}

The colors show the named contour plot's line colors and associated contour (Z) values and contour data units. All of the image plot's characteristics are represented, including color table, cindex, and fixed colors.

graphName can be just the name of a top-level graph window or a subwindow specification like Panel0#G0. See **Subwindow Syntax** on page III-92 for details on subwindow specifications.

contourFill=*contourInstanceName*

contourFill={*graphName*,*contourInstanceName*}

The colors show the named contour plot's fill colors and associated contour (Z) values and contour data units. All of the image plot's characteristics are represented, including color table, cindex, and fixed colors.

graphName can be just the name of a top-level graph window or a subwindow specification like Panel0#G0. See **Subwindow Syntax** on page III-92 for details on subwindow specifications.

The contourFill keyword was added in Igor Pro 8.00.

ctab={*zMin*,*zMax*,*ctName*, *reverse*}

The color table specified by `ctName` is drawn in the color legend.

zMin and *zMax* set the range of z values to map.

The color table name can be omitted if you want to leave it unchanged.

ctName can be any color table name returned by the CTabList function, such as Grays or Rainbow (see **Image Color Tables** on page II-392) or the name of a 3 column or 4 column color table wave (**Color Table Waves** on page II-399).

A color table wave name supplied to `ctab` must not be the name of a built-in color table (see **CTabList**). A 3 column or 4 column color table wave must have values that range between 0 and 65535. Column 0 is red, 1 is green, and 2 is blue. In column 3 a value of 65535 is fully opaque and 0 is fully transparent.

Set *reverse* to 1 to reverse the color table. Setting it to 0 or omitting it leaves the color table unreversed.

image=*imageInstanceName*

image={*graphName*,*imageInstanceName*}

The colors show the named image plot's colors and associated image (Z) values and image data units. All of the image plot's characteristics are represented, including color table, cindex, lookup wave, eval colors, and NaN transparency. **Note**: only false-color image plots can be used with ColorScale (see **Indexed Color Details** on page II-400).

graphName can be just the name of a top-level graph window or a subwindow specification like Panel0#G0. See **Subwindow Syntax** on page III-92 for details on subwindow specifications.

logLabel=*t*

t is the maximum number of decades before minor tick labels are suppressed. The default value is 3.

The logLabel keyword was added in Igor Pro 7.00.

logTicks=*t*

t=0 means "auto". This is the default value.

If *t* is not 0 then it represents the maximum number of decades before minor ticks are suppressed.

The logLabel keyword was added in Igor Pro 7.00.

lookup=*waveName*

Specifies an optional 1D wave used to modify the mapping of scaled Z values into the color table specified with the ctab keyword. Values should range from 0.0 to 1.0. A linear ramp from 0 to 1 would have no effect while a ramp from 1 to 0 would reverse the image. Used to apply gamma correction to grayscale images or for special effects. Use `lookup=$""` to remove option.

This keyword is not needed with the image keyword, even if the image plot uses a lookup wave. The image plot's lookup wave is used instead of the ColorScale lookup wave.

path=*gizmoObjectSpec*

path={*gizmoName*, *gizmoObjectSpec*}

The colors show the named Gizmo path plot's colors and associated X, Y, or Z values and units. Only the path plot's color table is represented. Other color modes such as color wave or solid colors are not supported.

gizmoName can be just the name of a top-level Gizmo window or a subwindow specification like Panel0#GZ0. See **Subwindow Syntax** on page III-92 for details on subwindow specifications.

gizmoObjectSpec can be just the name of the object like path0, or a colon-separated path to the object in a group like group0:path0. Unlike **ModifyGizmo** setCurrentGroup, *gizmoObjectSpec* does not start with the name of the Gizmo window.

The path keyword was added in Igor Pro 9.00.

ribbon=*gizmoObjectSpec*

ribbon={*gizmoName*, *gizmoObjectSpec*}

The colors show the named Gizmo ribbon plot's colors and associated X, Y, or Z values and units. Only the ribbon plot's color table is represented. Other color modes such as color wave or solid colors are not supported.

See the path keyword for {*gizmoName*, *gizmoObjectSpec*} details.

The ribbon keyword was added in Igor Pro 9.00.

scatter=*gizmoObjectSpec*

scatter={*gizmoName*, *gizmoObjectSpec*}

The colors show the named Gizmo scatter plot's marker colors and associated X, Y, or Z values and units. Only the scatter plot's color table is represented. Other color modes such as color wave or solid colors are not supported.

See the path keyword for {*gizmoName*, *gizmoObjectSpec*} details.

The scatter keyword was added in Igor Pro 9.00.

surface=*gizmoObjectSpec*

surface={*gizmoName*,*gizmoObjectSpec*}

The colors show the named Gizmo surface plot's grid line colors and associated surface X, Y, or Z values and surface matrix units. Unlike image plots, only the surface plot's color table is represented. Other color modes such as color wave or solid colors are not supported.

See the path keyword for {*gizmoName*, *gizmoObjectSpec*} details.

The surface keyword was added in Igor Pro 8.00.

surfaceFill=*gizmoObjectSpec*

surfaceFill={*gizmoName*,*gizmoObjectSpec*}

The colors show the named Gizmo surface plot's fill colors and associated surface X, Y, or Z values and surface matrix units. Unlike image plots, only the surface plot's color table is represented. Other color modes such as color wave or solid colors are not supported.

See the path keyword for {*gizmoName*, *gizmoObjectSpec*} details.

The surfaceFill keyword was added in Igor Pro 8.00.

trace=*traceInstanceName*

trace={*graphName*,*traceInstanceName*}

The colors show the color(s) of the named trace. This is useful when the trace has its color set by a "Z wave" using the `ModifyGraph zColor(`*`traceName`*`)=`... feature. In the Modify Trace Appearance dialog this is selected in the "Set as f(z)" subdialog. The color scale's main axis shows the range of values in the Z wave, and displays any data units the wave may have.

graphName can be just the name of a top-level graph window or a subwindow specification like Panel0#G0. See **Subwindow Syntax** on page III-92 for details on subwindow specifications.

Size Parameters

The following keyword-value parameters modify the size of the color scale annotation. These keywords are similar to those used by the **Slider** control. The size of the annotation is indirectly controlled by setting the

size of the "color bar" and the various axis parameters. The annotation sizes itself to accommodate the color bar, tick labels, and axis labels.

height=*h* Sets the height of the color bar in points, overriding any heightPct setting. The default height is 75% of the plot area height if the color scale is vertical, or a constant of 15 points if the color scale is horizontal. The default is restored by specifying `height=0`. Specifying a heightPct value resets height to this default.

heightPct=*hpct* Sets height as a percentage of the graph's plot area, overriding any height setting. The default height is 75% of the plot area height if the color scale is vertical, or a constant of 15 points if the color scale is horizontal. The default height is restored by setting heightPct=0. Specifying a height value resets heightPct to this default.

side=*s* Selects on which axis to draw main axis ticks.

- *s*=1: Right of the color bar if vert=1, or below if vert=0.
- *s*=2: Left of the color bar if vert=1, or above if vert=0.

vert=*v* Specifies color scale orientation.

- *v*=0: Horizontal.
- *v*=1: Vertical (default).

width=*w* Sets the width of the color bar in points, overriding any widthPct setting. The default width is a constant 15 points if the color scale is vertical, or 75% of the plot area width if the color scale is horizontal. The default is restored by specifying `width=0`. Specifying a widthPct value resets width to this default.

widthPct=*wpct* Sets width as a percentage of the graph's plot area, overriding any width setting. The default width is a constant 15 points if the color scale is vertical, or 75% of the plot area width if the color scale is horizontal. The default is restored by setting `widthPct=0`. Specifying a width value resets widthPct to this default.

Color Bar Parameters

The following keyword-value parameters modify the appearance of the color scale color bar.

colorBoxesFrame=*on* Draws frames surrounding up to 99 swatches of colors in the color bar (*on*=1).

When specifying more than 99 colors in the color bar (such as the Rainbow color table, which has 100 colors), the boxes aren't framed. Framing color boxes is effective only for small numbers of colors. Set the width of the frame with the frame keyword.

Use *on*=0 to turn off color box frames.

frame=*f* Specifies the thickness of the frame drawn around the color bar in points (*f* can range from 0 to 5 points). The default is 1 point. Fractional values are permitted.

Turn frames off with *f*=0. Values less that 0.5 do not display on screen, but the thin frame will print.

frameRGB=(*r*,*g*,*b*[,*a*]) or 0 Sets the color of the frame around the color bar. *r*, *g*, and *b* specify the amount of red, green, and blue as integers from 0 to 65535. The frame includes the individual color bar colors when colorBoxesFrame=1.

The frame will use the colorscale foreground color, as set by the /G flag, when frameRGB=0.

Axis Parameters

The following keyword-value parameters modify the appearance of the color scale axes. These keywords are based on the **ModifyGraph** Axis keywords because they modify the main or secondary color scale axes.

axisRange={*zMin*, *zMax*}

Sets the color bar axis range to values specified by *zMin* and *zMax*. Use * to use the default axis range for either or both values.

Omit *zMin* or *zMax* to leave that end of the range unchanged. For example, use `{zMin, }` to change *zMin* and leave *zMax* alone, or use `{ ,*}` to set only the axis maximum value to the default value.

dateInfo={*sd*,*tm*,*dt*}

Controls formatting of date/time axes.

sd=0:	Show date in the date&time format.
sd=1:	Suppress date.
sd=2:	Suppress time. The time is always shown if *tm*=2. *sd*=2 requires Igor Pro 9.00 or later.
tm=0:	12 hour (AM/PM) time.
tm=1:	24 hour (military) time.
tm=2:	Elapsed time.
dt=0:	Short dates (2/22/90).
dt=1:	Long dates (Thursday, February 22, 1990).
dt=2:	Abbreviated dates (Thurs, Feb 22, 1990).

These have no effect unless the axis is controlled by a wave with 'dat' data units.

These date formats do not work with dates before 0001-01-01 in which case the axis displays an empty string. You can apply custom date formats using the dateFormat keyword.

To use a custom date format as specified by the dateFormat keyword, you must specify -1 for the *dt* parameter to dateInfo.

For an f(z) color scale:

```
SetScale d, 0,0, "dat", fOfZWave
```

For a contour plot or image plot color scale:

```
SetScale d, 0,0, "dat", ZorXYZorImageWave
```

See **Date/Time Axes** on page II-315 and **Date, Time, and Date&Time Units** on page II-69 for details on how date/time axes work.

font=*fontNameStr*

Name of font as string expression. If the font does not exist, the default font is used. Specifying "default" has the same effect. Unlike ModifyGraph, the *fontNameStr* is evaluated at runtime, and its absence from the system is not an error.

fsize=*s*

Sets the font size in points.

s=0:	Use the graph font size for tick labels and axis labels (default).

fstyle=*fs*

Sets the font style. *fs* is a bitwise parameter with each bit controlling one aspect of the font style for the tick mark labels:

Bit 0:	Bold
Bit 1:	Italic
Bit 2:	Underline
Bit 4:	Strikethrough

See **Setting Bit Parameters** on page IV-12 for details about bit settings.

highTrip=*h*

If the extrema of an axis are between its lowTrip and its highTrip then tick mark labels use fixed point notation. Otherwise they use exponential (scientific or engineering) notation. The default highTrip is 100,000.

lblLatPos=*p* — Sets a lateral offset for the main axis label. This is an offset parallel to the axis. *p* is in points. Positive is down for vertical axes and to the right for horizontal axes. The default is 0.

lblMargin=*m* — Moves the main axis label by *m* points (default is 0) from the normal position. The default value is -5, which brings the axis label closer to the axis. Use more positive values to move the axis label away from the axis.

lblRot=*r* — Rotates the axis label by *r* degrees. *r* is a value from -360 to 360. Rotation is counterclockwise and starts from the label's normal orientation.

log=*l* — Specifies the axis type:

- *l*=0: Linear (default).
- *l*=1: Log base 10.
- *l*=2: Log base 2.

logHTrip=*h* — Same as highTrip but for log axes. The default is 10,000.

logLTrip=*l* — Same as lowTrip but for log axes. The default is 0.0001.

logTicks=*t* — Specifies the maximum number of decades in log axis before minor ticks are suppressed.

lowTrip=*l* — If the axis extrema are between its lowTrip and its highTrip, then tick mark labels use fixed point notation. Otherwise, they use exponential (scientific or engineering) notation. The default lowTrip is 0.1.

minor=*m* — Controls minor tick marks:

- *m*=0: Disables minor ticks (default).
- *m*=1: Enables minor ticks.

notation=*n* — Controls tick label notation:

- *n*=0: Engineering notation (default).
- *n*=1: Scientific notation.

nticks=*n* — Specifies the approximate number of ticks to be distributed along the main axis. Ticks are labelled using the same automatic algorithm used for graph axes. The default is 5.

Set *n*=0 for no ticks.

prescaleExp=*exp* — Multiplies axis range by 10*exp* for tick labeling and *exp* is subtracted from the axis label exponent. In other words, the exponent is moved from the tick labels to the axis label. The default is 0 (no modification). See the discussion in the **ModifyGraph (axes) Details** section.

tickExp=*te* — Controls tick label exponential notation:

- *te*=1: Forces tick labels to exponential notation when labels have units with a prefix.
- *te*=0: Turns off exponential notation.

tickLen=*t*

Sets the length of the ticks. *t* is the major tick mark length in points. This value must be between -100 and 50.

t= 0 to 50:	Draws tick marks between the tick labels and the colors box.
t= -1:	Default; auto tick length, equal to 70% of the tick label font size. Draws tick marks between the tick labels and the colors box.
t= -2 to -50:	Draws tick marks crossing the edge of the colors box nearest the tick labels. The actual total tick mark length is -*t*.
t= -100 to -51:	Draws tick marks inside the edge of the colors box nearest the tick labels. Actual tick mark length is -(*t*+50). For example, -58 makes in an inside tick mark that is 8 points long.

tickThick=*t*

Sets the tick mark thickness in points (from 0 to 5 points). The default is 1 point. Fractional values are permitted.

t=0: Turns tick marks off, but not the tick labels.

tickUnit=*tu*

Turns on (*tu*=0) or off (*tu*=1) units labels attached to tick marks.

userTicks={*tvWave*,*tlblWave*}

Supplies user defined tick positions and labels for the main axis. *tvWave* contains the numeric tick positions while text wave *tlblWave* contains the corresponding labels.

Overrides normal ticking specified by nticks.

See **User Ticks from Waves** on page II-313 for details.

The tick mark labels can be multiline and use styled text. For more details, see **Fancy Tick Mark Labels** on page II-358.

ZisZ=*z*

z =1 labels the zero tick mark (if any) with the single digit "0" regardless of the number of digits used for other labels. Default is *z*=0.

Secondary Axis Parameters

axisLabel2=*axisLabelString2*

Axis label for the secondary axis. This axis label is drawn only if userTicks2 is in effect. Text after any \r character is ignored, as is the \r character. The default is "".

lblLatPos2=*p*

Sets lateral offset for secondary axis labels. This is an offset parallel to the axis. *p* is in points. Positive is down for vertical axes and to the right for horizontal axes. The default is 0.

lblMargin2=*m*

Specifies the distance in points (default 0) to move the secondary axis label from the position that would be normal for a graph. The default is value is -5, which brings the axis label closer to the axis. Use more positive values to move the axis label away from the axis.

lblRot2=*r*

Rotates the secondary axis label by *r* degrees counterclockwise starting from the normal label orientation. r is a value from -360 to 360.

userTicks2={*tvWave*,*tlblWave*}

Supplies user defined tick positions and labels for a second axis which is always on the opposite side of the color bar from the main axis. The tick mark labels can be multiline and use styled text. For more details, see **Fancy Tick Mark Labels** on page II-358. This is the only way to draw a second axis.