**RBC Test Cases**

**Graph Component**

**Authors:**

Sam Green

Nick Hudson

Stanton Sievers

Jarrod Stormo

# Table of Contents

[Table of Contents 2](#_Toc216945757)

[Graph Bar 4](#_Toc216945758)

[Graph Configure 4](#_Toc216945759)

[Graph Crosshairs Cget 15](#_Toc216945760)

[Graph Crosshairs Configure 15](#_Toc216945761)

[Graph Crosshairs Off 19](#_Toc216945762)

[Graph Crosshairs On 19](#_Toc216945763)

[Graph Crosshairs Toggle 20](#_Toc216945764)

[Graph Extents 20](#_Toc216945765)

[Graph Element Cget 22](#_Toc216945766)

[Graph Element Cget 23](#_Toc216945767)

[Graph Element Configure 23](#_Toc216945768)

[Graph Element Activate 38](#_Toc216945769)

[Graph Element Bind 39](#_Toc216945770)

[Graph Element Closest 39](#_Toc216945771)

[Graph Element Deactivate 41](#_Toc216945772)

[Graph Element Delete 42](#_Toc216945773)

[Graph Element Exists 42](#_Toc216945774)

[Graph Element Names 43](#_Toc216945775)

[Graph Element Show 43](#_Toc216945776)

[Graph Element Type 44](#_Toc216945777)

[Graph Grid Cget 45](#_Toc216945778)

[Graph Grid Configure Color 45](#_Toc216945779)

[Graph Grid Off 50](#_Toc216945780)

[Graph Grid On 51](#_Toc216945781)

[Graph Grid Toggle 51](#_Toc216945782)

[Graph Inside 52](#_Toc216945783)

[Graph Invtransform 53](#_Toc216945784)

[Graph Legend 53](#_Toc216945785)

[Graph Line 73](#_Toc216945786)

[Graph Marker 73](#_Toc216945787)

[Graph Pen 85](#_Toc216945788)

[Graph PostScript 96](#_Toc216945789)

[Graph Snap 101](#_Toc216945790)

[Graph Transform 102](#_Toc216945791)

[Graph Xaxis 102](#_Toc216945792)

[Graph X2axis 102](#_Toc216945793)

[Graph Yaxis 103](#_Toc216945794)

[Graph Y2axis 103](#_Toc216945795)

# Graph Bar

### Test Case 1

**Test Case ID –** RBC.graph.bar.1

**Test Item –** The *bar* function of the *graph* BLT component.

**Input Specification –** The bar data to show in the graph

**Output Specification –** A bar graph element is added to the graph

**Special Procedural Requirements –** The operations on this data are tested in Barchart test cases

**Inter-case Dependencies –** None

# Graph Configure

### Test Case 1

**Test Case ID –** RBC.graph.configure.1

**Test Item –** The *configure* function of the *graph* BLT component.

**Input Specification –** A name value pair for the configuration options

**Output Specification –** The configuration options being changed on the graph

**Special Procedural Requirements –** None

**Inter-case Dependencies –** None

# Graph Crosshairs Cget

### Test Case 1

**Test Case ID –** RBC.graph.crosshairs.cget.1

**Test Item –** The *crosshairs cget* function of the *graph* BLT component.

**Input Specification –** The name of a crosshairs configure option.

**Output Specification –** The value of the given crosshairs configure option name.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –**

# Graph Crosshairs Configure

### Test Case 1

**Test Case ID –** RBC.graph.crosshairs.configure.1

**Test Item –** The *crosshairs configure -color* function of the *graph* BLT component.

**Input Specification –** A string representation of a color name.

**Output Specification –** The color of the crosshairs component is set to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.crosshairs.cget.1

### Test Case 2

**Test Case ID –** RBC.graph.crosshairs.configure.2

**Test Item –** The *crosshairs configure -dashes* function of the *graph* BLT component.

**Input Specification –** A list of up to 11 numbers that alternately represent the lengths of the dashes and gaps on the cross hair lines.

**Output Specification –** The dash style of the crosshairs component is set to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.crosshairs.cget.1

### Test Case 3

**Test Case ID –** RBC.graph.crosshairs.configure.3

**Test Item –** The *crosshairs configure -hide* function of the *graph* BLT component.

**Input Specification –** Any of the following: 1, 0, true, false, yes, no

**Output Specification –** The hide property of the crosshairs component is set according to the input value (either 1, 0, True, False, Yes, or No).

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.crosshairs.cget.1

### Test Case 4

**Test Case ID –** RBC.graph.crosshairs.configure.4

**Test Item –** The *crosshairs configure -linewidth* function of the *graph* BLT component.

**Input Specification –** A positive numerical value.

**Output Specification –** The linewidth of the crosshairs component is set according to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.crosshairs.cget.1

### Test Case 5

**Test Case ID –** RBC.graph.crosshairs.configure.6

**Test Item –** The *crosshairs configure -position* function of the *graph* BLT component.

**Input Specification –** Window coordinates in the form “@x,y”.

**Output Specification –** The coordinates of the crosshair intersection is set to the given coordinates.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.crosshairs.cget.1

# Graph Crosshairs Off

### Test Case 1

**Test Case ID –** RBC.graph.crosshairs.off.1

**Test Item –** The *crosshairs off* command of the *graph* BLT component.

**Input Specification –** None

**Output Specification –** The crosshairs on the graph component should be hidden.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** None

# Graph Crosshairs On

### Test Case 1

**Test Case ID –** RBC.graph.crosshairs.on.1

**Test Item –** The *crosshairs on* command of the *graph* BLT component.

**Input Specification –** None

**Output Specification –** The crosshairs on the graph component should be visible.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** None

# Graph Crosshairs Toggle

### Test Case 1

**Test Case ID –** RBC.graph.crosshairs.toggle.1

**Test Item –** The *crosshairs toggle* command of the *graph* BLT component.

**Input Specification –** None

**Output Specification –** The visibility of the crosshairs on the graph component should be the opposite of what it was to start.

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.crosshairs.off.1, RBC.graph.crosshairs.on.1

# Graph Extents

### Test Case 1

**Test Case ID –** RBC.graph.extents.1

**Test Item –** The *extents* function of the *graph* BLT component.

**Input Specification –** The margin to show extents for

**Output Specification –** The size of the margin or plot dimension

**Special Procedural Requirements -** None

**Inter-case Dependencies –** None

# Graph Element Cget

### Test Case 1

**Test Case ID –** RBC.graph.element.cget.1

**Test Item –** The *element cget* function of the *graph* BLT component.

**Input Specification –** The name of a element configure option.

**Output Specification –** The value of the given element configure option name.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Cget: Valid Option Name

**Test Case 1**

**Purpose –** Ensure the *element cget* command works correctly when given a valid element configuration option name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.cget.1.1

### Test Procedure – Graph Element Cget: Invalid Option Name

**Test Case 1**

**Purpose –** Ensure the *crosshairs cget* command works correctly when given an invalid element configuration option name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.cget.1.2

# Graph Element Create

### Test Case 1

**Test Case ID –** RBC.graph.element.create.1

**Test Item –** The *element create* function of the *graph* BLT component.

**Input Specification –** A unique name for the element.

**Output Specification –** A new element that can be referred to by the input name.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.names.1

### Test Procedure – Graph Element Create: Unique Name

**Purpose –** Ensure the *element create* command works correctly when given a unique element name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.create.1.1

### Test Procedure – Graph Element Create: Existing Name

**Purpose –** Ensure the *element create* command works correctly when given an existing element name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.create.1.2

# Graph Element Configure

### Test Case 1

**Test Case ID –** RBC.graph.element.configure.1

**Test Item –** The *element configure -activepen* function of the *graph* BLT component.

**Input Specification –** The name of a pen or “” to not draw the element when it is active.

**Output Specification –** The activepen property of the element component is set to the input value, which will be used to draw the element when it is active.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.cget.1,RBC.graph.element.create.1, RBC.graph.pen.1

### Test Procedure – Graph Element Configure: Activepen – Valid Pen Name

**Purpose –** Ensure the *element configure -activepen* command works correctly when given a valid pen name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.1.1

### Test Procedure – Graph Element Configure: Activepen – Invalid Pen Name

**Purpose –** Ensure the *element configure -activepen* command works correctly when given an invalid pen name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.1.2

### Test Case 2

**Test Case ID –** RBC.graph.element.configure.2

**Test Item –** The *element configure -bindtags* function of the *graph* BLT component.

**Input Specification –** A list of binding tag names.

**Output Specification –** The bindtag list has the given list of tag names.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.cget.1, RBC.graph.element.create.1, RBC.graph.element.bind.1

### Test Procedure – Graph Element Configure: Bindtags – Single Tag

**Purpose –** Ensure the *element configure -bindtags* command works correctly when given a single tag name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.2.1

### Test Procedure – Graph Element Configure: Bindtags – Tag List

**Purpose –** Ensure the *element configure -bindtags* command works correctly when given a list of tag names.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.2.2

### Test Case 3

**Test Case ID –** RBC.graph.element.configure.3

**Test Item –** The *element configure -color* function of the *graph* BLT component.

**Input Specification –** A string representation of a color name.

**Output Specification –** The color of the element component is set to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.cget.1, RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Color – Valid Color Name

**Purpose –** Ensure the *element configure -color* command works correctly when given a valid color name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.3.1

### Test Procedure – Graph Element Configure: Color – Invalid Color Name

**Purpose –** Ensure the *element configure -color* command works correctly when given an invalid color name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.3.2

### Test Case 4

**Test Case ID –** RBC.graph.element.configure.4

**Test Item –** The *element configure -dashes* function of the *graph* BLT component.

**Input Specification –** A list of up to 11 numbers that alternately represent the lengths of the dashes and gaps between the elements.

**Output Specification –** The dash style of the element component is set to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.cget.1, RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Dashes – Valid Dash List

**Purpose –** Ensure the *element configure -dashes* command works correctly when given a valid dash list.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.4.1

### Test Procedure – Graph Element Configure: Dashes – Empty Dash List

**Purpose –** Ensure the *element configure -dashes* command works correctly when given an empty dash list.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.4.2

### Test Procedure – Graph Element Configure: Dashes – Long Dash List

**Purpose –** Ensure the *element configure -dashes* command works correctly when given a dash list that is too long.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.4.3

### Test Procedure – Graph Element Configure: Dashes – Invalid Numerical Dash List

**Purpose –** Ensure the *element configure -dashes* command works correctly when given an invalid numerical dash list.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.4.4

### Test Procedure – Graph Element Configure: Dashes – Dash List with Characters

**Purpose –** Ensure the *element configure -dashes* command works correctly when given a dash list with characters.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.4.5

### Test Case 5

**Test Case ID –** RBC.graph.element.configure.5

**Test Item –** The *element configure -data* function of the *graph* BLT component.

**Input Specification –** A list of numeric expressions representing the X-Y coordinate pairs of each data point.

**Output Specification –** The data property of the element is set to the list of input.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.cget.1,RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Data – Valid Coordinate List

**Purpose –** Ensure the *element configure -data* command works correctly when given a valid coordinate list as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.5.1

### Test Procedure – Graph Element Configure: Data – Invalid Coordinate List

**Purpose –** Ensure the *element configure -data* command works correctly when given an invalid coordinate list as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.5.2

### Test Procedure – Graph Element Configure: Data – Not Proper Form

**Purpose –** Ensure the *element configure -data* command works correctly when given input that is not in the proper form (x1 y1 x2 y1).

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.5.3

### Test Procedure – Graph Element Configure: Data – Odd List

**Purpose –** Ensure the *element configure -data* command works correctly when given input that does not have an even number of values.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.5.4

### Test Case 6

**Test Case ID –** RBC.graph.element.configure.6

**Test Item –** The *element configure -fill* function of the *graph* BLT component.

**Input Specification –** A string representation of a color name, defcolor, or “”.

**Output Specification –** The fill of the element component is set to the input color name, the fill color will use the -color option if given defcolor, or the fill color will be transparent if given “”.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.cget.1, RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Fill – Valid Color Name

**Purpose –** Ensure the *element configure -fill* command works correctly when given a valid color name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.6.1

### Test Procedure – Graph Element Configure: Fill – Invalid Color Name

**Purpose –** Ensure the *element configure -fill* command works correctly when given an invalid color name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.6.2

### Test Procedure – Graph Element Configure: Fill – Defcolor

**Purpose –** Ensure the *element configure -fill* command works correctly when given ‘defcolor’ as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.6.3

### Test Procedure – Graph Element Configure: Fill – Empty String

**Purpose –** Ensure the *element configure -fill* command works correctly when given “” as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.6.4

### Test Case 7

**Test Case ID –** RBC.graph.element.configure.7

**Test Item –** The *element configure -hide* function of the *graph* BLT component.

**Input Specification –** Any of the following: 1, 0, true, false, yes, no

**Output Specification –** The hide property of the element component is set according to the input value (either 1 or 0).

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Hide – 1

**Purpose –** Ensure the *element configure -hide* command works correctly when given 1.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.7.1

### Test Procedure – Graph Element Configure: Hide – 0

**Purpose –** Ensure the *element configure -hide* command works correctly when given 0.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.7.2

### Test Procedure – Graph Element Configure: Hide – True

**Purpose –** Ensure the *element configure -hide* command works correctly when given true.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.7.3

### Test Procedure – Graph Element Configure: Hide – False

**Purpose –** Ensure the *element configure -hide* command works correctly when given false.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.7.4

### Test Procedure – Graph Element Configure: Hide – Yes

**Purpose –** Ensure the *element configure -hide* command works correctly when given yes.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.7.5

### Test Procedure – Graph Element Configure: Hide – No

**Purpose –** Ensure the *element configure -hide* command works correctly when given no.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.7.6

### Test Procedure – Graph Element Configure: Hide – Invalid Input

**Purpose –** Ensure the *element configure -hide* command works correctly when given an invalid input value.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.7.7

### Test Case 8

**Test Case ID –** RBC.graph.element.configure.8

**Test Item –** The *element configure -label* function of the *graph* BLT component.

**Input Specification –** Text label for the element or “” for no legend entry.

**Output Specification –** The label of the element is set to the input value and is displayed in the legend or no entry is displayed in the legend and the value of the label is empty.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.cget.1, RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Label – Valid Text

**Purpose –** Ensure the *element configure -label* command works correctly when given valid text.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.8.1

### Test Procedure – Graph Element Configure: Label – Empty String

**Purpose –** Ensure the *element configure -label* command works correctly when given the empty string as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.8.2

### Test Procedure – Graph Element Configure: Label – Default Value

**Purpose –** Ensure the *element configure -label* command does not change the default value when not given any input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.8.3

### Test Case 9

**Test Case ID –** RBC.graph.element.configure.9

**Test Item –** The *element configure -linewidth* function of the *graph* BLT component.

**Input Specification –** A positive numerical value.

**Output Specification –** The linewidth of the element component is set according to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Linewidth – Valid Integer Input

**Purpose –** Ensure the *element configure -linewidth* command works correctly when given an integer pixel value.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.9.1

### Test Procedure – Graph Element Configure: Linewidth – Valid Decimal Input

**Purpose –** Ensure the *element configure -linewidth* command works correctly when given a decimal pixel value.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.9.2

### Test Procedure – Graph Element Configure: Linewidth – Invalid Numerical Input

**Purpose –** Ensure the *element configure -linewidth* command works correctly when given an invalid numerical pixel value.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.9.3

### Test Procedure – Graph Element Configure: Linewidth – Character Input

**Purpose –** Ensure the *element configure -linewidth* command works correctly when given a character as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.9.4

### Test Case 10

**Test Case ID –** RBC.graph.element.configure.10

**Test Item –** The *element configure -mapx* function of the *graph* BLT component.

**Input Specification –** The name of a graph axis instance.

**Output Specification –** The element’s x-coordinates are mapped onto the given axis.

**Special Procedural Requirements –** A graph widget can be created, an axis component can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1, RBC.graph.axis.[create]

### Test Procedure – Graph Element Configure: Mapx – Valid Axis Name

**Purpose –** Ensure the *element configure -mapx* command works correctly when given valid axis name as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.10.1

### Test Procedure – Graph Element Configure: Mapx – Non-Existent Axis Name

**Purpose –** Ensure the *element configure -mapx* command works correctly when given a non-existent axis name as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.10.2

### Test Procedure – Graph Element Configure: Mapx – No Input

**Purpose –** Ensure the *element configure -mapx* command works correctly when not input is given.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.10.3

### Test Case 11

**Test Case ID –** RBC.graph.element.configure.11

**Test Item –** The *element configure -mapy* function of the *graph* BLT component.

**Input Specification –** The name of a graph axis instance.

**Output Specification –** The element’s y-coordinates are mapped onto the given axis.

**Special Procedural Requirements –** A graph widget can be created, an axis component can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1, RBC.graph.axis.[create]

### Test Procedure – Graph Element Configure: Mapy – Valid Axis Name

**Purpose –** Ensure the *element configure -mapy* command works correctly when given valid axis name as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.11.1

### Test Procedure – Graph Element Configure: Mapy – Non-Existent Axis Name

**Purpose –** Ensure the *element configure -mapy* command works correctly when given a non-existent axis name as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.11.2

### Test Procedure – Graph Element Configure: Mapy – No Input

**Purpose –** Ensure the *element configure -mapy* command works correctly when not input is given.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.11.3

### Test Case 12

**Test Case ID –** RBC.graph.element.configure.12

**Test Item –** The *element configure -offdash* function of the *graph* BLT component.

**Input Specification –** A string representation of a color name, defcolor, or “”.

**Output Specification –** The offdash of the element component is set to the input color name, the offdash color will use the -color option if given defcolor, or the offdash pixels will represent gaps instead of stripes if given “”.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.cget.1, RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Offdash – Valid Color Name

**Purpose –** Ensure the *element configure -offdash* command works correctly when given a valid color name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.12.1

### Test Procedure – Graph Element Configure: Offdash – Invalid Color Name

**Purpose –** Ensure the *element configure -offdash* command works correctly when given an invalid color name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.12.2

### Test Procedure – Graph Element Configure: Offdash – Defcolor

**Purpose –** Ensure the *element configure -offdash* command works correctly when given ‘defcolor’ as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.12.3

### Test Procedure – Graph Element Configure: Offdash – Empty String

**Purpose –** Ensure the *element configure -offdash* command works correctly when given “” as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.12.4

### Test Case 13

**Test Case ID –** RBC.graph.element.configure.13

**Test Item –** The *element configure -outline* function of the *graph* BLT component.

**Input Specification –** A string representation of a color name, defcolor, or “”.

**Output Specification –** The outline of the element component is set to the input color name, the outline color will use the -color option if given defcolor, or the outline will not be drawn if given “”.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.cget.1, RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Outline – Valid Color Name

**Purpose –** Ensure the *element configure -outline* command works correctly when given a valid color name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.13.1

### Test Procedure – Graph Element Configure: Outline – Invalid Color Name

**Purpose –** Ensure the *element configure -outline* command works correctly when given an invalid color name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.13.2

### Test Procedure – Graph Element Configure: Outline – Defcolor

**Purpose –** Ensure the *element configure -outline* command works correctly when given ‘defcolor’ as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.13.3

### Test Case 14

**Test Case ID –** RBC.graph.element.configure.14

**Test Item –** The *element configure -pen* function of the *graph* BLT component.

**Input Specification –** The name of a pen.

**Output Specification –** The pen property of the element component is set to the input value, which will be used to draw the element when it is active.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.cget.1,RBC.graph.element.create.1, RBC.graph.pen.1

### Test Procedure – Graph Element Configure: Pen – Valid Pen Name

**Purpose –** Ensure the *element configure -pen* command works correctly when given a valid pen name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.14.1

### Test Procedure – Graph Element Configure: Pen – Invalid Pen Name

**Purpose –** Ensure the *element configure -pen* command works correctly when given an invalid pen name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.14.2

### Test Case 15

**Test Case ID –** RBC.graph.element.configure.15

**Test Item –** The *element configure -outlinewidth* function of the *graph* BLT component.

**Input Specification –** A positive numerical value.

**Output Specification –** The outlinewidth of the element component is set according to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Outlinewidth – Valid Integer Input

**Purpose –** Ensure the *element configure -outlinewidth* command works correctly when given an integer pixel value.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.15.1

### Test Procedure – Graph Element Configure: Outlinewidth – Valid Decimal Input

**Purpose –** Ensure the *element configure -outlinewidth* command works correctly when given a decimal pixel value.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.15.2

### Test Procedure – Graph Element Configure: Outlinewidth – Invalid Numerical Input

**Purpose –** Ensure the *element configure -outlinewidth* command works correctly when given an invalid numerical pixel value.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.15.3

### Test Procedure – Graph Element Configure: Outlinewidth – Character Input

**Purpose –** Ensure the *element configure -outlinewidth* command works correctly when given a character as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.15.4

### Test Case 16

**Test Case ID –** RBC.graph.element.configure.16

**Test Item –** The *element configure -scalesymbols* function of the *graph* BLT component.

**Input Specification –** Any of the following: 1, 0, true, false, yes, no

**Output Specification –** The scalessymbols property of the element component is set according to the input value (either 1 or 0).

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Scalesymbols – 1

**Purpose –** Ensure the *element configure -scalesymbols* command works correctly when given 1.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.16.1

### Test Procedure – Graph Element Configure: Scalesymbols – 0

**Purpose –** Ensure the *element configure -scalesymbols* command works correctly when given 0.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.16.2

### Test Procedure – Graph Element Configure: Scalesymbols – True

**Purpose –** Ensure the *element configure -scalesymbols* command works correctly when given true.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.16.3

### Test Procedure – Graph Element Configure: Scalesymbols – False

**Purpose –** Ensure the *element configure -scalesymbols* command works correctly when given false.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.16.4

### Test Procedure – Graph Element Configure: Scalesymbols – Yes

**Purpose –** Ensure the *element configure -scalesymbols* command works correctly when given yes.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.16.5

### Test Procedure – Graph Element Configure: Scalesymbols – No

**Purpose –** Ensure the *element configure -scalesymbols* command works correctly when given no.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.16.6

### Test Procedure – Graph Element Configure: Scalesymbols – Invalid Input

**Purpose –** Ensure the *element configure -scalesymbols* command works correctly when given an invalid input value.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.16.7

### Test Case 17

**Test Case ID –** RBC.graph.element.configure.17

**Test Item –** The *element configure -smooth* function of the *graph* BLT component.

**Input Specification –** Any of the following: linear, step, natural, quadratic

**Output Specification –** The smooth property of the element component is set to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Smooth – Valid Input

**Purpose –** Ensure the *element configure -smooth* command works correctly when given valid input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.17.1

### Test Procedure – Graph Element: Smooth – Invalid Input

**Purpose –** Ensure the *element configure -smooth* command works correctly when given invalid input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.17.2

### Test Case 18

**Test Case ID –** RBC.graph.element.configure.18

**Test Item –** The *element configure -styles* function of the *graph* BLT component.

**Input Specification –** A list containing a pen name and optionally a minimum and maximum element weight range.

**Output Specification –** The styles property of the element component is set to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element: Stylelist – Valid Pen Name

**Purpose –** Ensure the *element configure -styles* command works correctly when given a valid pen name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.18.1

### Test Procedure – Graph Element: Stylelist – Non-Existent Pen

**Purpose –** Ensure the *element configure -styles* command works correctly when given an invalid pen name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.18.2

### Test Procedure – Graph Element Configure: Stylelist – Pen and Weights

**Purpose –** Ensure the *element configure -styles* command works correctly when given a valid pen name and a minimum and maximum element weight range.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.18.3

### Test Case 19

**Test Case ID –** RBC.graph.element.configure.19

**Test Item –** The *element configure -symbol* function of the *graph* BLT component.

**Input Specification –** Any of the following: square, circle, diamond, plus, cross, splus, scross, triangle, “”, bitmap.

**Output Specification –** The symbol property of the element component is set to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Symbol – Valid Symbol Name

**Purpose –** Ensure the *element configure -symbol* command works correctly when given a valid symbol name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.19.1

### Test Procedure – Graph Element Configure: Symbol – Invalid Symbol Name

**Purpose –** Ensure the *element configure -symbol* command works correctly when given an invalid symbol name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.19.2

### Test Case 20

**Test Case ID –** RBC.graph.element.configure.20

**Test Item –** The *element configure -trace* function of the *graph* BLT component.

**Input Specification –** Any of the following: increasing, decreasing, both

**Output Specification –** The trace property of the element component is set to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Trace – Valid Direction

**Purpose –** Ensure the *element configure -trace* command works correctly when given a valid direction.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.20.1

### Test Procedure – Graph Element Configure: Trace – Invalid Direction

**Purpose –** Ensure the *element configure -trace* command works correctly when given an invalid direction.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.20.2

### Test Case 21

**Test Case ID –** RBC.graph.element.configure.21

**Test Item –** The *element configure -weights* function of the *graph* BLT component.

**Input Specification –** A vector name or a list of numeric expressions representing the weights for each data point.

**Output Specification –** The weights property of the element component is set to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Weights – Valid Vector Name

**Purpose –** Ensure the *element configure -weights* command works correctly when given a valid vector name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.21.1

### Test Procedure – Graph Element Configure: Weights – Invalid Vector Name

**Purpose –** Ensure the *element configure -weights* command works correctly when given an invalid vector name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.21.2

### Test Procedure – Graph Element Configure: Weights – Valid List

**Purpose –** Ensure the *element configure -weights* command works correctly when given a valid list of numerical values.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.21.3

### Test Procedure – Graph Element Configure: Weights – Invalid Input

**Purpose –** Ensure the *element configure -weights* command works correctly when given an invalid list.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.21.4

### Test Case 22

**Test Case ID –** RBC.graph.element.configure.22

**Test Item –** The *element configure -xdata* function of the *graph* BLT component.

**Input Specification –** A vector name or a list of numeric expressions.

**Output Specification –** The xdata property of the element component is set to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Xdata – Valid Vector Name

**Purpose –** Ensure the *element configure -xdata* command works correctly when given a valid vector name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.22.1

### Test Procedure – Graph Element Configure: Xdata – Invalid Vector Name

**Purpose –** Ensure the *element configure -xdata* command works correctly when given an invalid vector name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.22.2

### Test Procedure – Graph Element Configure: Xdata – Valid List

**Purpose –** Ensure the *element configure -xdata* command works correctly when given a valid list of numerical values.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.22.3

### Test Procedure – Graph Element Configure: Xdata – Invalid Input

**Purpose –** Ensure the *element configure -xdata* command works correctly when given an invalid list.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.22.4

### Test Case 23

**Test Case ID –** RBC.graph.element.configure.23

**Test Item –** The *element configure -ydata* function of the *graph* BLT component.

**Input Specification –** A vector name or a list of numeric expressions.

**Output Specification –** The ydata property of the element component is set to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Ydata – Valid Vector Name

**Purpose –** Ensure the *element configure -ydata* command works correctly when given a valid vector name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.23.1

### Test Procedure – Graph Element Configure: Ydata – Invalid Vector Name

**Purpose –** Ensure the *element configure -ydata* command works correctly when given an invalid vector name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.23.2

### Test Procedure – Graph Element Configure: Ydata – Valid List

**Purpose –** Ensure the *element configure -ydata* command works correctly when given a valid list of numerical values.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.23.3

### Test Procedure – Graph Element Configure: Ydata – Invalid Input

**Purpose –** Ensure the *element configure -ydata* command works correctly when given an invalid list.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.23.4

### Test Case 24

**Test Case ID –** RBC.graph.element.configure.24

**Test Item –** The *element configure -pixels* function of the *graph* BLT component.

**Input Specification –** A positive number.

**Output Specification –** The pixels property of the element component is set to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Configure: Pixels – Positive Integer

**Purpose –** Ensure the *element configure -pixels* command works correctly when given a positive integer.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.24.1

### Test Procedure – Graph Element Configure: Pixels – Positive Decimal

**Purpose –** Ensure the *element configure -pixels* command works correctly when given a positive decimal.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.24.2

### Test Procedure – Graph Element Configure: Pixels – Invalid Numerical Input

**Purpose –** Ensure the *element configure -pixels* command works correctly when given invalid numerical input (e.g. a negative value).

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.24.3

### Test Procedure – Graph Element Configure: Pixels – Character Input

**Purpose –** Ensure the *element configure -pixels* command works correctly when given characters as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.24.4

# Graph Element Activate

### Test Case 1

**Test Case ID –** RBC.graph.element.activate.1

**Test Item –** The *element activate* function of the *graph* BLT component.

**Input Specification –** An element name.

**Output Specification –** The element name is on the activate list.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Activate: Valid Element Name

**Purpose –** Ensure the *element activate* command works correctly when given a valid element name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.activate.1.1

### Test Procedure – Graph Element Activate: Invalid Element Name

**Purpose –** Ensure the *element activate* command works correctly when given an invalid element name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.activate.1.2

# Graph Element Bind

### Test Case 1

**Test Case ID –** RBC.graph.element.bind.1

**Test Item –** The *element bind* function of the *graph* BLT component.

**Input Specification –** A tag name and a command.

**Output Specification –** The bindtags contain the name of the tag.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Bind: Tag, Sequence, Command

**Purpose –** Ensure the *element bind* command works correctly when given a tag name, an event sequence, and a command.

**Special Requirements –** None

**TclTest –** RBC.graph.element.bind.1.1

# Graph Element Closest

### Test Case 1

**Test Case ID –** RBC.graph.element.closest.1

**Test Item –** The *element closest* function of the *graph* BLT component.

**Input Specification –** An x and y screen coordinate and a variable name in which the results will be stored.

**Output Specification –** The variable contains the name of the closest element, the index of the closest data points, and the graph coordinates of the point. The function returns 0 if no data point within the threshold is found, otherwise it returns 1.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Closest: Existing Closest Element

**Purpose –** Ensure the *element closest* command works correctly when a closest element exists.

**Special Requirements –** None

**TclTest –** RBC.graph.element.closest.1.1

### Test Procedure – Graph Element Closest: Existing Closest Element

**Purpose –** Ensure the *element closest* command works correctly when a closest element does not exist.

**Special Requirements –** None

**TclTest –** RBC.graph.element.closest.1.2

### Test Case 2

**Test Case ID –** RBC.graph.element.closest.2

**Test Item –** The *element closest* function of the *graph* BLT component with the *-along* flag.

**Input Specification –** An x and y screen coordinate, a variable name in which the results will be stored, and a direction to the *-along* flag (x, y, both).

**Output Specification –** The variable contains the name of the closest element, the index of the closest data points, and the graph coordinates of the point. The function returns 0 if no data point within the threshold is found, otherwise it returns 1.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Closest: Along – X

**Purpose –** Ensure the *element closest -along* command works correctly when given x as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.closest.2.1

### Test Procedure – Graph Element Closest: Along – Y

**Purpose –** Ensure the *element closest -along* command works correctly when given y as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.closest.2.2

### Test Procedure – Graph Element Closest: Along – Both

**Purpose –** Ensure the *element closest -along* command works correctly when given both as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.closest.2.3

### Test Procedure – Graph Element Closest: Along – Invalid Input

**Purpose –** Ensure the *element closest -along* command works correctly when given invalid input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.closest.2.4

### Test Case 3

**Test Case ID –** RBC.graph.element.closest.3

**Test Item –** The *element closest* function of the *graph* BLT component with the *-halo* flag.

**Input Specification –** An x and y screen coordinate, a variable name in which the results will be stored, and a pixel value for the *-halo* flag.

**Output Specification –** The variable contains the name of the closest element, the index of the closest data points, and the graph coordinates of the point. The function returns 0 if no data point within the threshold is found, otherwise it returns 1.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Closest: Along – Positive Integer Value

**Purpose –** Ensure the *element closest -halo* command works correctly when given a positive integer value as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.closest.3.1

### Test Case 4

**Test Case ID –** RBC.graph.element.closest.4

**Test Item –** The *element closest* function of the *graph* BLT component with the *-interpolate* flag.

**Input Specification –** An x and y screen coordinate, a variable name in which the results will be stored, and any of the following to the *-along* flag: 1, 0, true, false, yes, no

**Output Specification –** The variable contains the name of the closest element, the index of the closest data points, and the graph coordinates of the point. The function returns 0 if no data point within the threshold is found, otherwise it returns 1.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Closest: Along – 1

**Purpose –** Ensure the *element closest -along* command works correctly when given 1 as input.

**Special Requirements –** None

**TclTest –** RBC.graph.element.configure.11.1

# Graph Element Create

### Test Case 1

**Test Case ID –** RBC.graph.element.create.1

**Test Item –** The *element create* function of the *graph* BLT component.

**Input Specification –** An element name.

**Output Specification –** A graph element will be created

**Special Procedural Requirements –** A graph widget can be created

**Inter-case dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Create: Unique Element Name

**Purpose –** Ensure the element create command works correctly when given a unique element name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.create.1.1

### Test Procedure – Graph Element Create: Existing Element Name

**Purpose –** Ensure the element create command works correctly when given an existing element name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.type.1.2

# Graph Element Deactivate

### Test Case 1

**Test Case ID –** RBC.graph.element.deactivate.1

**Test Item –** The *element deactivate* function of the *graph* BLT component.

**Input Specification –** An element name.

**Output Specification –** The element name is not on the activate list.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1, RBC.graph.element.activate.1

### Test Procedure – Graph Element Deactivate: Valid Element Name

**Purpose –** Ensure the *element deactivate* command works correctly when given a valid element name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.deactivate.1.1

### Test Procedure – Graph Element Deactivate: Invalid Element Name

**Purpose –** Ensure the *element deactivate* command works correctly when given an invalid element name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.deactivate.1.2

# Graph Element Delete

### Test Case 1

**Test Case ID –** RBC.graph.element.delete.1

**Test Item –** The *element delete* function of the *graph* BLT component.

**Input Specification –** One or more element name.

**Output Specification –** The element name is deleted.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Delete: Single Element Name

**Purpose –** Ensure the *element delete* command works correctly when given a single element name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.delete.1.1

### Test Procedure – Graph Element Delete: Multiple Element Names

**Purpose –** Ensure the *element delete* command works correctly when given multiple element names.

**Special Requirements –** None

**TclTest –** RBC.graph.element.delete.1.2

### Test Procedure – Graph Element Delete: Invalid Element Name

**Purpose –** Ensure the *element delete* command works correctly when given an invalid element name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.delete.1.3

# Graph Element Exists

### Test Case 1

**Test Case ID –** RBC.graph.element.exists.1

**Test Item –** The *element exists* function of the *graph* BLT component.

**Input Specification –** An element name.

**Output Specification –** 1 if the element is found, 0 otherwise.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.element.create.1, RBC.graph.element.activate.1

### Test Procedure – Graph Element Exists: Existing Element Name

**Purpose –** Ensure the *element delete* command works correctly when given an existing element name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.exists.1.1

### Test Procedure – Graph Element Exists: Non-Existent Element Name

**Purpose –** Ensure the *element delete* command works correctly when given a non-existent element name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.exists.1.2

# Graph Element Names

### Test Case 1

**Test Case ID –** RBC.graph.element.names.1

**Test Item –** The *element names* function of the *graph* BLT component.

**Input Specification –** A pattern.

**Output Specification –** The names of element objects that match the pattern.

**Special Procedural Requirements –** A graph widget can be created

**Inter-case dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Names: No Pattern

**Purpose –** Ensure the *element names* command works correctly when no pattern is given.

**Special Requirements –** None

**TclTest –** RBC.graph.element.names.1.1

### Test Procedure – Graph Element Names: Exact Pattern

**Purpose –** Ensure the *element names* command works correctly when an exact pattern is given.

**Special Requirements –** None

**TclTest –** RBC.graph.element.names.1.2

### Test Procedure – Graph Element Names: Wildcard Pattern

**Purpose –** Ensure the *element names* command works correctly when a wildcard pattern is given.

**Special Requirements –** None

**TclTest –** RBC.graph.element.names.1.3

### Test Procedure – Graph Element Names: Incorrect Pattern

**Purpose –** Ensure the *element names* command works correctly when an incorrect pattern is given.

**Special Requirements –** None

**TclTest –** RBC.graph.element.names.1.4

# Graph Element Show

### Test Case 1

**Test Case ID –** RBC.graph.element.show.1

**Test Item –** The *element show* function of the *graph* BLT component.

**Input Specification –** One or more element names.

**Output Specification –** The given element names are part of the show list.

**Special Procedural Requirements –** A graph widget can be created

**Inter-case dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Show: Single Element Name

**Purpose –** Ensure the *element show* command works correctly when given a single element name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.show.1.1

### Test Procedure – Graph Element Show: Multiple Element Names

**Purpose –** Ensure the *element show* command works correctly when given multiple element names.

**Special Requirements –** None

**TclTest –** RBC.graph.element.show.1.2

### Test Procedure – Graph Element Show: Non-Existent Element Name

**Purpose –** Ensure the *element show* command works correctly when given a non-existent element name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.show.1.3

# Graph Element Type

### Test Case 1

**Test Case ID –** RBC.graph.element.type.1

**Test Item –** The *element type* function of the *graph* BLT component.

**Input Specification –** An element name.

**Output Specification –** One of the following: BarElement, LineElement.

**Special Procedural Requirements –** A graph widget can be created

**Inter-case dependencies –** RBC.graph.element.create.1

### Test Procedure – Graph Element Type: Valid Line Element Name

**Purpose –** Ensure the *element type* command works correctly when given the name of a valid line element.

**Special Requirements –** None

**TclTest –** RBC.graph.element.type.1.1

### Test Procedure – Graph Element Type: Valid Bar Element Name

**Purpose –** Ensure the *element type* command works correctly when given the name of a valid bar element.

**Special Requirements –** None

**TclTest –** RBC.graph.element.type.1.2

### Test Procedure – Graph Element Type: Invalid Element Name

**Purpose –** Ensure the *element type* command works correctly when given an invalid element name.

**Special Requirements –** None

**TclTest –** RBC.graph.element.type.1.3

# Graph Grid Cget

### Test Case 1

**Test Case ID –** RBC.graph.grid.1

**Test Item –** The *grid cget* function of the *graph* BLT component.

**Input Specification –** The name of a grid configure option.

**Output Specification –** The value of the given grid configure option name.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –**

### Test Procedure – Graph Grid Cget: Valid Option Name

**Purpose –** Ensure the *grid cget* command works correctly when given a valid grid configuration option name.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.1.1

### Test Procedure – Graph Grid Cget: Invalid Option Name

**Purpose –** Ensure the *grid cget* command works correctly when given an invalid grid configuration option name.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.1.2

# Graph Grid Configure Color

### Test Case 1

**Test Case ID –** RBC.graph.grid.1

**Test Item –** The *grid configure -color* function of the *graph* BLT component.

**Input Specification –** A string representation of a color name.

**Output Specification –** The color of the grid component is set to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.grid.1

### Test Procedure – Graph Grid Configure: Color – Valid Color Name

**Purpose –** Ensure the *grid configure -color* command works correctly when given a valid color name.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.1.1

### Test Procedure – Graph Grid Configure: Color – Invalid Color Name

**Purpose –** Ensure the *grid configure -color* command works correctly when given an invalid color name.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.1.2

### Test Case 2

**Test Case ID –** RBC.graph.grid.2

**Test Item –** The *grid configure -dashes* function of the *graph* BLT component.

**Input Specification –** A list of up to 11 numbers that alternately represent the lengths of the dashes and gaps on the cross hair lines.

**Output Specification –** The dash style of the grid component is set to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.grid.1

### Test Procedure – Graph Grid Configure: Dashes – Valid Dash List

**Purpose –** Ensure the *grid configure -dashes* command works correctly when given a valid dash list.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.2.1

### Test Procedure – Graph Grid Configure: Color – Empty Dash List

**Purpose –** Ensure the *grid configure -dashes* command works correctly when given an empty dash list.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.2.2

### Test Procedure – Graph Grid Configure: Dashes – Long Dash List

**Purpose –** Ensure the *grid configure -dashes* command works correctly when given a dash list that is too long.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.2.3

### Test Procedure – Graph Grid Configure: Dashes – Invalid Numerical Dash List

**Purpose –** Ensure the *grid configure -dashes* command works correctly when given an invalid numerical dash list.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.2.4

### Test Procedure – Graph Grid Configure: Dashes – Dash List with Characters

**Purpose –** Ensure the *grid configure -dashes* command works correctly when given a dash list with characters.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.2.5

### Test Case 3

**Test Case ID –** RBC.graph.grid.configure.3

**Test Item –** The *grid configure -hide* function of the *graph* BLT component.

**Input Specification –** Any of the following: 1, 0, true, false, yes, no

**Output Specification –** The hide property of the grid component is set according to the input value (either 1 or 0).

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.grid.cget.1

### Test Procedure – Graph Grid Configure: Hide – 1

**Purpose –** Ensure the *grid configure -hide* command works correctly when given 1.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.3.1

### Test Procedure – Graph Grid Configure: Hide – 0

**Purpose –** Ensure the *grid configure -hide* command works correctly when given 0.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.3.2

### Test Procedure – Graph Grid Configure: Hide – True

**Purpose –** Ensure the *grid configure -hide* command works correctly when given true.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.3.3

### Test Procedure – Graph Grid Configure: Hide – False

**Purpose –** Ensure the *grid configure -hide* command works correctly when given false.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.3.4

### Test Procedure – Graph Grid Configure: Hide – Yes

**Purpose –** Ensure the *grid configure -hide* command works correctly when given yes.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.3.5

### Test Procedure – Graph Grid Configure: Hide – No

**Purpose –** Ensure the *grid configure -hide* command works correctly when given no.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.3.6

### Test Procedure – Graph Grid Configure: Hide – Invalid Input

**Purpose –** Ensure the *grid configure -hide* command works correctly when given an invalid input value.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.3.7

### Test Case 4

**Test Case ID –** RBC.graph.grid.configure.4

**Test Item –** The *grid configure -linewidth* function of the *graph* BLT component.

**Input Specification –** A positive numerical value.

**Output Specification –** The linewidth of the grid component is set according to the input value.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.grid.1

### Test Procedure – Graph Grid Configure: Linewidth – Valid Integer Input

**Purpose –** Ensure the *grid configure -linewidth* command works correctly when given an integer pixel value.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.4.1

### Test Procedure – Graph Grid Configure: Linewidth – Valid Decimal Input

**Purpose –** Ensure the *grid configure -linewidth* command works correctly when given a decimal pixel value.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.4.2

### Test Procedure – Graph Grid Configure: Linewidth – Invalid Numerical Input

**Purpose –** Ensure the *grid configure -linewidth* command works correctly when given an invalid numerical pixel value.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.4.3

### Test Procedure – Graph Grid Configure: Linewidth – Character Input

**Purpose –** Ensure the *grid configure -linewidth* command works correctly when given a character as input.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.4.4

### Test Case 5

**Test Case ID –** RBC.graph.grid.configure.5

**Test Item –** The *grid configure -mapx* function of the *graph* BLT component.

**Input Specification –** The name of a graph axis instance or “” for no grid lines.

**Output Specification –** The x-axis of the grid is set to the given axis or no grid lines are displayed.

**Special Procedural Requirements –** A graph widget can be created, an axis component can be created.

**Inter-case Dependencies –** RBC.graph.grid.1, RBC.graph.axis.[create]

### Test Procedure – Graph Grid Configure: Mapx – Valid Axis Name

**Purpose –** Ensure the *grid configure -mapx* command works correctly when given valid axis name as input.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.5.1

### Test Procedure – Graph Grid Configure: Mapx – Non-Existent Axis Name

**Purpose –** Ensure the *grid configure -mapx* command works correctly when given a non-existent axis name as input.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.5.2

### Test Procedure – Graph Grid Configure: Mapx – Empty String

**Purpose –** Ensure the *grid configure -mapx* command works correctly when given the empty string as input.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.5.3

### Test Procedure – Graph Grid Configure: Mapx – No Input

**Purpose –** Ensure the *grid configure -mapx* command works correctly when not input is given.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.5.4

### Test Case 6

**Test Case ID –** RBC.graph.grid.configure.6

**Test Item –** The *grid configure -mapy* function of the *graph* BLT component.

**Input Specification –** The name of a graph axis instance or “” for no grid lines.

**Output Specification –** The y-axis of the grid is set to the given axis or no grid lines are displayed.

**Special Procedural Requirements –** A graph widget can be created, an axis component can be created.

**Inter-case Dependencies –** RBC.graph.grid.1, RBC.graph.axis.[create]

### Test Procedure – Graph Grid Configure: Mapy – Valid Axis Name

**Purpose –** Ensure the *grid configure -mapy* command works correctly when given valid axis name as input.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.6.1

### Test Procedure – Graph Grid Configure: Mapy – Non-Existent Axis Name

**Purpose –** Ensure the *grid configure -mapy* command works correctly when given a non-existent axis name as input.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.6.2

### Test Procedure – Graph Grid Configure: Mapy – Empty String

**Purpose –** Ensure the *grid configure -mapy* command works correctly when given the empty string as input.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.6.3

### Test Procedure – Graph Grid Configure: Mapy – No Input

**Purpose –** Ensure the *grid configure -mapy* command works correctly when not input is given.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.6.4

### Test Case 7

**Test Case ID –** RBC.graph.grid.configure.7

**Test Item –** The *grid configure -minor* function of the *graph* BLT component.

**Input Specification –** Any of the following: 1, 0, true, false, yes, no

**Output Specification –** The minor property of the grid component is set according to the input value (either 1 or 0).

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.grid.1

### Test Procedure – Graph Grid Configure: Minor – 1

**Purpose –** Ensure the *grid configure -minor* command works correctly when given 1.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.7.1

### Test Procedure – Graph Grid Configure: Minor – 0

**Purpose –** Ensure the *grid configure -minor* command works correctly when given 0.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.7.2

### Test Procedure – Graph Grid Configure: Minor – True

**Purpose –** Ensure the *grid configure -minor* command works correctly when given true.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.7.3

### Test Procedure – Graph Grid Configure: Minor – False

**Purpose –** Ensure the *grid configure -minor* command works correctly when given false.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.7.4

### Test Procedure – Graph Grid Configure: Minor – Yes

**Purpose –** Ensure the *grid configure -minor* command works correctly when given yes.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.7.5

### Test Procedure – Graph Grid Configure: Minor – No

**Purpose –** Ensure the *grid configure -minor* command works correctly when given no.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.7.6

### Test Procedure – Graph Grid Configure: Minor – Invalid Input

**Purpose –** Ensure the *grid configure -minor* command works correctly when given an invalid input value.

**Special Requirements –** None

**TclTest –** RBC.graph.grid.configure.7.7

# Graph Grid Off

### Test Case 1

**Test Case ID –** RBC.graph.grid.off.1

**Test Item –** The *grid off* command of the *graph* BLT component.

**Input Specification –** None

**Output Specification –** The grid on the graph component should be hidden.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** None

### Test Procedure – Graph Grid Off

**Purpose –** Ensure the grid can be hidden on a graph.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.grid.off.1.tcl” file and then call the “graph.grid::RBC.graph.grid.off.1.1.Setup” Tcl command
* Pre-Condition – There is a graph with the grid currently displaying
* Body

1. Call the “graph.grid::RBC.graph.grid.off.1.1.Body” Tcl command

* Post-Condition – The grid on the graph is hidden
* Cleanup – Call the “graph.grid::RBC.graph.grid.off.1.1.Cleanup” command

# Graph Grid On

### Test Case 1

**Test Case ID –** RBC.graph.grid.on.1

**Test Item –** The *grid on* command of the *graph* BLT component.

**Input Specification –** None

**Output Specification –** The grid on the graph component should be visible.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** None

### Test Procedure – Graph Grid On

**Purpose –** Ensure the grid can be displayed on a graph.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.grid.on.1.tcl” file and then call the “graph.grid::RBC.graph.grid.on.1.1.Setup” Tcl command
* Pre-Condition – There is a graph with the grid currently hidden
* Body

1. Call the “graph.grid::RBC.graph.grid.on.1.1.Body” Tcl command

* Post-Condition – The grid on the graph is visible
* Cleanup – Call the “graph.grid::RBC.graph.grid.on.1.1.Cleanup” command

# Graph Grid Toggle

### Test Case 1

**Test Case ID –** RBC.graph.grid.toggle.1

**Test Item –** The *grid toggle* command of the *graph* BLT component.

**Input Specification –** None

**Output Specification –** The visibility of the grid on the graph component should be the opposite of what it was to start.

**Special Procedural Requirements –** A graph widget can be created.

**Inter-case Dependencies –** RBC.graph.grid.off.1, RBC.graph.grid.on.1

### Test Procedure – Graph Grid Toggle: Off

**Purpose –** Ensure the grid can be toggled on a graph.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.grid.toggle.1.tcl” file and then call the “graph.grid::RBC.graph.grid.toggle.1.1.Setup” Tcl command
* Pre-Condition – There is a graph with the grid currently displaying
* Body

1. Call the “graph.grid::RBC.graph.grid.toggle.1.1.Body” Tcl command

* Post-Condition – The grid on the graph is hidden
* Cleanup – Call the “graph.grid::RBC.graph.grid.toggle.1.1.Cleanup” command

### Test Procedure – Graph Grid Toggle: On

**Purpose –** Ensure the grid can be toggled on a graph.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.grid.toggle.1.tcl” file and then call the “graph.grid::RBC.graph.grid.toggle.1.2.Setup” Tcl command
* Pre-Condition – There is a graph with the grid currently hidden
* Body

1. Call the “graph.grid::RBC.graph.grid.toggle.1.2.Body” Tcl command

* Post-Condition – The grid on the graph is visible
* Cleanup – Call the “graph.grid::RBC.graph.grid.toggle.1.2.Cleanup” command

# Graph Inside

### Test Case 1

**Test Case ID –** RBC.graph.inside.1

**Test Item –** The *inside* function of the *graph* BLT component.

**Input Specification –** A screen coordinate (X and Y)

**Output Specification –** Whether the screen coordinate is in the plot area

**Special Procedural Requirements –** None

**Inter-case Dependencies –** None

### Test Procedure – Inside: Not Inside

**Purpose –** Ensure that inside returns 0 for when it is outside

**Special Requirements –** None

**Script –** RBC.graph.inside.1.tcl

**Procedural Steps**

* Setup – Call the “graph.inside::RBC.graph.inside.1.1.Setup” Tcl command
* Pre-Condition – A graph is displayed
* Body

1. Call the “graph.inside::RBC.graph.inside.1.1.Body” Tcl command

* Post-Condition – Command returns 0
* Cleanup – Call the “graph.inside::RBC.graph.inside.1.1.Cleanup” command

### Test Procedure – Inside: Inside

**Purpose –** Ensure that inside returns 1 for when it is inside the plot.

**Special Requirements –** None

**Script –** RBC.graph.inside.1.tcl

**Procedural Steps**

* Setup – Call the “graph.inside::RBC.graph.inside.1.2.Setup” Tcl command
* Pre-Condition – A graph is displayed
* Body

1. Call the “graph.inside::RBC.graph.inside.1.2.Body” Tcl command

* Post-Condition – Command returns 1
* Cleanup – Call the “graph.inside::inside.graph.inside.1.2.Cleanup” command

# Graph Invtransform

### Test Case 1

**Test Case ID –** RBC.graph.invtransform.1

**Test Item –** The *invtransform* function of the *graph* BLT component.

**Input Specification –** A window coordinate (X and Y)

**Output Specification –** The graph coordinates translated from the inputs

**Special Procedural Requirements –** None

**Inter-case Dependencies –** None

### Test Procedure – Invtransform

**Purpose –** Ensure the invtransform command works correctly.

**Special Requirements –** None

**Script –** RBC.graph.invtransform.1.tcl

**Procedural Steps**

* Setup – Call the “graph.invtransform::RBC.graph.invtransform.1.1.Setup” Tcl command
* Pre-Condition – A graph is displayed
* Body

1. Call the “graph.invtransform::RBC.graph.invtransform.1.1.Body” Tcl command

* Post-Condition – Command returns -0.0765550239234 0.972392638037
* Cleanup – Call the “graph.invtransform::RBC.graph.invtransform.1.1.Cleanup” command

# Graph Legend

### Test Case 1

**Test Case ID –** RBC.graph.legend.1

**Test Item –** The *legend activate* command of the *graph* BLT component.

**Input Specification –** A legend element.

**Output Specification –** The legend element should appear on the list of activated elements.

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Activate All Returns All Activated Elements

**Purpose –** Ensure that activating all legend elements returns all legend elements.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.1.1

### Test Procedure – Activate Returns Subset of Activated Elements

**Purpose –** Ensure that activating a subset of legend elements returns only the activated elements.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.1.2

### Test Procedure – Activate Returns All Activated Elements

**Purpose –** Ensure that activating a legend element returns all active legend elements not only the ones that were just activated.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.1.3

### Test Case 2

**Test Case ID –** RBC.graph.legend.2

**Test Item –** The *legend deactivate* command of the *graph* BLT component.

**Input Specification –** A legend element.

**Output Specification –** The legend element should not appear on the list of activated elements.

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.element.create.1, RBC.graph.legend.1

### Test Procedure – Deactivate All Elements

**Purpose –** Ensure that deactivating all legend elements works properly.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.2.1

### Test Procedure – Deactivate Subset of Elements

**Purpose –** Ensure that deactivating a subset of legend elements deactivates only that subset.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.2.2

### Test Case 3

**Test Case ID –** RBC.graph.legend.3

**Test Item –** The *legend configure* command of the *graph* BLT component.

**Input Specification –** A valid configuration *option* flag and *value* pair

**Output Specification –** *legend cget* *-option* should return *value*

**Special Procedural Requirements –** None

**Inter-case Dependencies -** RBC.graph.legend.5

### Test Procedure – Configure Active Background

**Purpose –** Ensure that the activebackground configuration works for valid colors.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.1

### Test Procedure – Configure Active Border Width

**Purpose –** Ensure that the activeborderwidth configuration works for valid widths.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.2

### Test Procedure – Configure Active Foreground

**Purpose –** Ensure that the activeforeground configuration works for valid colors.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.3

### Test Procedure – Configure Active Relief Raised

**Purpose –** Ensure that the activerelief configuration works for raised reliefs.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.4

### Test Procedure – Configure Active Relief Flat

**Purpose –** Ensure that the activerelief configuration works for flat reliefs.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.5

### Test Procedure – Configure Active Relief Grooved

**Purpose –** Ensure that the activerelief configuration works for grooved reliefs.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.6

### Test Procedure – Configure Active Relief Ridged

**Purpose –** Ensure that the activerelief configuration works for ridged reliefs.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.7

### Test Procedure – Configure Active Relief Solid

**Purpose –** Ensure that the activerelief configuration works for solid reliefs.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.8

### Test Procedure – Configure Active Relief Sunken

**Purpose –** Ensure that the activerelief configuration works for sunken reliefs.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.9

### Test Procedure – Configure Anchor Center

**Purpose –** Ensure that the anchor configuration works for center anchor.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.10

### Test Procedure – Configure Anchor North

**Purpose –** Ensure that the anchor configuration works for north anchor.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.11

### Test Procedure – Configure Anchor Northeast

**Purpose –** Ensure that the anchor configuration works for northeast anchor.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.12

### Test Procedure – Configure Anchor Northwest

**Purpose –** Ensure that the anchor configuration works for northwest anchor.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.13

### Test Procedure – Configure Anchor South

**Purpose –** Ensure that the anchor configuration works for south anchor.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.14

### Test Procedure – Configure Anchor Southeast

**Purpose –** Ensure that the anchor configuration works for southeast anchor.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.15

### Test Procedure – Configure Anchor Southwest

**Purpose –** Ensure that the anchor configuration works for southwest anchor.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.16

### Test Procedure – Configure Anchor East

**Purpose –** Ensure that the anchor configuration works for east anchor.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.17

### Test Procedure – Configure Anchor West

**Purpose –** Ensure that the anchor configuration works for west anchor.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.18

### Test Procedure – Configure No Background

**Purpose –** Ensure that the background configuration works for no background.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.19

### Test Procedure – Configure Background

**Purpose –** Ensure that the background configuration works for valid colors.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.20

### Test Procedure – Configure Borderwidth

**Purpose –** Ensure that the borderwidth configuration works for valid width.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.21

### Test Procedure – Configure Font

**Purpose –** Ensure that the font configuration works for valid font strings.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.22

### Test Procedure – Configure Foreground

**Purpose –** Ensure that the foreground configuration works for valid colors.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.23

### Test Procedure – Configure Hidden

**Purpose –** Ensure that the hide configuration works for true.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.24

### Test Procedure – Configure Not Hidden

**Purpose –** Ensure that the hide configuration works for false.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.25

### Test Procedure – Configure Ipadx Single

**Purpose –** Ensure that the ipadx configuration works for a single value.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.26

### Test Procedure – Configure Ipadx Multiple

**Purpose –** Ensure that the ipadx configuration works for multiple values.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.27

### Test Procedure – Configure Ipady Single

**Purpose –** Ensure that the ipady configuration works for a single value.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.28

### Test Procedure – Configure Ipady Multiple

**Purpose –** Ensure that the ipady configuration works for multiple values.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.29

### Test Procedure – Configure Padx Single

**Purpose –** Ensure that the padx configuration works for a single value.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.30

### Test Procedure – Configure Padx Multiple

**Purpose –** Ensure that the padx configuration works for multiple values.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.31

### Test Procedure – Configure Pady Single

**Purpose –** Ensure that the pady configuration works for a single value.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.32

### Test Procedure – Configure Pady Multiple

**Purpose –** Ensure that the pady configuration works for multiple values.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.33

### Test Procedure – Configure Position Right Margin

**Purpose –** Ensure that the position configuration works for rightmargin.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.34

### Test Procedure – Configure Position Left Margin

**Purpose –** Ensure that the position configuration works for leftmargin.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.35

### Test Procedure – Configure Position Top Margin

**Purpose –** Ensure that the position configuration works for topmargin.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.36

### Test Procedure – Configure Position Bottom Margin

**Purpose –** Ensure that the position configuration works for bottommargin.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.37

### Test Procedure – Configure Position Plot Area

**Purpose –** Ensure that the position configuration works for plotarea.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.38

### Test Procedure – Configure Position a Point

**Purpose –** Ensure that the position configuration works for a point.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.39

### Test Procedure – Configure Raised

**Purpose –** Ensure that the raised configuration works for true.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.40

### Test Procedure – Configure Not Raised

**Purpose –** Ensure that the raised configuration works for false.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.41

### Test Procedure – Configure Relief Raised

**Purpose –** Ensure that the relief configuration works for raised reliefs.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.42

### Test Procedure – Configure Relief Flat

**Purpose –** Ensure that the relief configuration works for flat reliefs.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.43

### Test Procedure – Configure Relief Grooved

**Purpose –** Ensure that the relief configuration works for grooved reliefs.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.44

### Test Procedure – Configure Relief Ridged

**Purpose –** Ensure that the relief configuration works for ridged reliefs.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.45

### Test Procedure – Configure Relief Solid

**Purpose –** Ensure that the relief configuration works for solid reliefs.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.46

### Test Procedure – Configure Relief Sunken

**Purpose –** Ensure that the relief configuration works for sunken reliefs.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.47

### Test Procedure – Configure Shadow

**Purpose –** Ensure that the shadow configuration works for valid colors.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.48

### Test Procedure – Configure Shadow and Depth

**Purpose –** Ensure that the shadow configuration works for a shadow and a depth.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.49

### Test Procedure – Configure No Shadow

**Purpose –** Ensure that the shadow configuration works for no shadow.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.3.50

### Test Case 4

**Test Case ID –** RBC.graph.legend.4

**Test Item –** The *legend bind* command of the *graph* BLT component.

**Input Specification –** A legend element or arbitrary tag name, an action sequence, and a command

**Output Specification –** *legend bind* should return the bound actions and commands

**Special Procedural Requirements –** None

**Inter-case Dependencies -** RBC.graph.element.create.1

### Test Procedure – Bind Legend Element

**Purpose –** Ensure that bindings can be created for a legend element.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.4.1

### Test Procedure – Bind Query for Sequence and Element

**Purpose –** Ensure that bindings can be queried for a sequence and legend element.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.4.2

### Test Procedure – Bind Command Append

**Purpose –** Ensure that bindings can be appended for a sequence and legend element.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.4.3

### Test Procedure – Bind Command Overwrite

**Purpose –** Ensure that bindings are overwritten for a sequence and legend element.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.4.4

### Test Procedure – Bind Query for Element

**Purpose –** Ensure that bound sequences can be queried for just a legend element.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.4.5

### Test Procedure – Bind Arbitrary Tag Name

**Purpose –** Ensure that bindings can be created for an arbitrary tag.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.4.6

### Test Case 5

**Test Case ID –** RBC.graph.legend.5

**Test Item –** The *legend cget* command of the *graph* BLT component.

**Input Specification –** A configuration *option* flag

**Output Specification –** The current value for the *option* flag

**Special Procedural Requirements –** None

**Inter-case Dependencies -** RBC.graph.legend.3

### Test Procedure – Cget Default Option

**Purpose –** Ensure that cget works for default values.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.5.1

### Test Procedure – Cget Configured Option

**Purpose –** Ensure that cget works with an explicitly set option.

**Special Requirements –** None

**TclTest –** RBC.graph.legend.5.2

### Test Case 6

**Test Case ID –** RBC.graph.legend.6

**Test Item –** The *legend activate* command of the *graph* BLT component.

**Input Specification –** A legend element.

**Output Specification –** The legend element should appear activated in the legend.

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.element.create.1

### Test Procedure – Activate All

**Purpose –** Ensure that activating all legend elements works properly.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.6.tcl” file and then call the “graph.legend::RBC.graph.legend.6.1.Setup” Tcl command
* Pre-Condition – A graph with two elements, Line1 and Line2, is showing. The two elements appear deactivated in the legend.
* Body

1. Call the “graph.legend::RBC.graph.legend.6.1.Body” Tcl command

* Post-Condition – In the legend, Line1 and Line2 appear activated (i.e. their color has changed to dark gray).
* Cleanup – Call the “graph.legend::RBC.graph.legend.6.1.Cleanup” command

### Test Procedure – Activate a Subset

**Purpose –** Ensure that activating a subset of legend elements works properly.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.6.tcl” file and then call the “graph.legend::RBC.graph.legend.6.2.Setup” Tcl command
* Pre-Condition – A graph with two elements, Line1 and Line2, is showing. The two elements appear deactivated in the legend.
* Body

1. Call the “graph.legend::RBC.graph.legend.6.2.Body” Tcl command

* Post-Condition – In the legend, Line1 and only Line1 appears activated (i.e. its color has changed to dark gray).
* Cleanup – Call the “graph.legend::RBC.graph.legend.6.2.Cleanup” command

### Test Case 7

**Test Case ID –** RBC.graph.legend.7

**Test Item –** The *legend deactivate* command of the *graph* BLT component.

**Input Specification –** A legend element.

**Output Specification –** The legend element should appear activated in the legend.

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.element.create.1, RBC.graph.legend.1

### Test Procedure – Deactivate All

**Purpose –** Ensure that deactivating all legend elements works properly.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.7.tcl” file and then call the “graph.legend::RBC.graph.legend.7.1.Setup” Tcl command
* Pre-Condition – A graph with two elements, Line1 and Line2, is showing. The two elements appear activated in the legend.
* Body

1. Call the “graph.legend::RBC.graph.legend.7.1.Body” Tcl command

* Post-Condition – In the legend, Line1 and Line2 appear deactivated (i.e. their color has changed to light gray).
* Cleanup – Call the “graph.legend::RBC.graph.legend.7.1.Cleanup” command

### Test Procedure – Deactivate a Subset

**Purpose –** Ensure that deactivating a subset of legend elements works properly.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.7.tcl” file and then call the “graph.legend::RBC.graph.legend.7.2.Setup” Tcl command
* Pre-Condition – A graph with two elements, Line1 and Line2, is showing. The two elements appear activated in the legend.
* Body

1. Call the “graph.legend::RBC.graph.legend.7.2.Body” Tcl command

* Post-Condition – In the legend, Line1 and only Line1 appears deactivated (i.e. its color has changed to light gray).
* Cleanup – Call the “graph.legend::RBC.graph.legend.7.2.Cleanup” command

### Test Case 8

**Test Case ID –** RBC.graph.legend.8

**Test Item –** The *legend configure* command of the *graph* BLT component.

**Input Specification –** A valid configuration *option* flag and *value* pair

**Output Specification –** The legend and its elements should reflect the new option values.

**Special Procedural Requirements –** None

**Inter-case Dependencies -** RBC.graph.element.1, RBC.graph.legend.6

### Test Procedure – Configure Active Background

**Purpose –** Ensure that deactivating a subset of legend elements works properly.

**Special Requirements –** Legend elements must be able to be activated.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.1.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing and the element in the legend is activated.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.1.Body” Tcl command

* Post-Condition – The legend element’s background is now a salmon color.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.1.Cleanup” command

### Test Procedure – Configure Active Border Width

**Purpose –** Ensure that the activeborderwidth configuration works for valid widths.

**Special Requirements –** Legend elements must be able to be activated.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.2.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing and the element in the legend is activated.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.2.Body” Tcl command

* Post-Condition – The legend element’s border is at a width of 20 pixels.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.2.Cleanup” command

### Test Procedure – Configure Active Foreground

**Purpose –** Ensure that the activeforeground configuration works for valid colors.

**Special Requirements –** Legend elements must be able to be activated.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.3.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing and the element in the legend is activated.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.3.Body” Tcl command

* Post-Condition – The legend element’s text is now white.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.3.Cleanup” command

### Test Procedure – Configure Active Relief Raised

**Purpose –** Ensure that the activerelief configuration works for raised reliefs.

**Special Requirements –** Legend elements must be able to be activated. Activeborderwidth must be able to be set.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.4.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing and the element in the legend is activated. The legend element should also have an active border width of 10.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.4.Body” Tcl command

* Post-Condition – The legend element’s relief is now raised.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.4.Cleanup” command

### Test Procedure – Configure Active Relief Flat

**Purpose –** Ensure that the activerelief configuration works for flat reliefs.

**Special Requirements –** Legend elements must be able to be activated. Activeborderwidth must be able to be set.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.5.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing and the element in the legend is activated. The legend element should also have active border width of 10.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.5.Body” Tcl command

* Post-Condition – The legend element’s relief is now flat.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.5.Cleanup” command

### Test Procedure – Configure Active Relief Grooved

**Purpose –** Ensure that the activerelief configuration works for grooved reliefs.

**Special Requirements –** Legend elements must be able to be activated. Activeborderwidth must be able to be set.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.6.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing and the element in the legend is activated. The legend element should also have an active border width of 10.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.6.Body” Tcl command

* Post-Condition – The legend element’s relief is now grooved.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.6.Cleanup” command

### Test Procedure – Configure Active Relief Ridged

**Purpose –** Ensure that the activerelief configuration works for ridged reliefs.

**Special Requirements –** Legend elements must be able to be activated. Activeborderwidth must be able to be set.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.7.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing and the element in the legend is activated. The legend element should also have an active border width of 10.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.7.Body” Tcl command

* Post-Condition – The legend element’s relief is now ridged.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.7.Cleanup” command

### Test Procedure – Configure Active Relief Solid

**Purpose –** Ensure that the activerelief configuration works for solid reliefs.

**Special Requirements –** Legend elements must be able to be activated. Activeborderwidth must be able to be set.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.8.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing and the element in the legend is activated. The legend element should also have an active border width of 10.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.8.Body” Tcl command

* Post-Condition – The legend element’s relief is now solid.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.8.Cleanup” command

### Test Procedure – Configure Active Relief Sunken

**Purpose –** Ensure that the activerelief configuration works for sunken reliefs.

**Special Requirements –** Legend elements must be able to be activated. Activeborderwidth must be able to be set.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.9.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing and the element in the legend is activated. The legend element should also have an active border width of 10.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.9.Body” Tcl command

* Post-Condition – The legend element’s relief is now sunken.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.9.Cleanup” command

### Test Procedure – Configure All Anchors

**Purpose –** Ensure that the anchor configuration works for all anchors.

**Special Requirements –** The legend must be able to be positioned in the plot area. As the bodies are executed, take note of the position of the legend.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.10.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing. The legend is in the top center of the plot area.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.10.Body1” Tcl command
2. Call the “graph.legend::RBC.graph.legend.8.10.Body2” Tcl command
3. Call the “graph.legend::RBC.graph.legend.8.10.Body3” Tcl command
4. Call the “graph.legend::RBC.graph.legend.8.10.Body4” Tcl command
5. Call the “graph.legend::RBC.graph.legend.8.10.Body5” Tcl command
6. Call the “graph.legend::RBC.graph.legend.8.10.Body6” Tcl command
7. Call the “graph.legend::RBC.graph.legend.8.10.Body7” Tcl command
8. Call the “graph.legend::RBC.graph.legend.8.10.Body8” Tcl command
9. Call the “graph.legend::RBC.graph.legend.8.10.Body9” Tcl command

* Post-Condition – The legend moved around the plot area in the following order:
  1. Center
  2. North Center
  3. Northeast
  4. East Center
  5. Southeast
  6. South Center
  7. Southwest
  8. West Center
  9. Northwest
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.10.Cleanup” command

### Test Procedure – Configure No Background

**Purpose –** Ensure that the background configuration works for no background.

**Special Requirements –** Legend background must be able to be set for valid colors

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.11.Setup” Tcl command
* Pre-Condition – A graph with a single element is displaying. The graph’s background color is salmon and the legend’s background is gray.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.11.Body” Tcl command

* Post-Condition – The legend’s background is now salmon.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.11.Cleanup” command

### Test Procedure – Configure Valid Background Color

**Purpose –** Ensure that the background configuration works for valid colors.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.12.Setup” Tcl command
* Pre-Condition – A graph with a single element is displaying. The legend’s background is gray.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.12.Body” Tcl command

* Post-Condition – The legend’s background is now black.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.12.Cleanup” command

### Test Procedure – Configure Border Width

**Purpose –** Ensure that the borderwidth configuration works for valid widths.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.13.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.13.Body” Tcl command

* Post-Condition – The legend element’s border is at a width of 20 pixels.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.13.Cleanup” command

### Test Procedure – Configure Font

**Purpose –** Ensure Ensure that the font configuration works for valid font strings.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.14.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.14.Body” Tcl command

* Post-Condition – The legend element’s border is now Arial, bold, 14 point.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.14.Cleanup” command

### Test Procedure – Configure Foreground

**Purpose –** Ensure that the foreground configuration works for valid colors.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.15.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.15.Body” Tcl command

* Post-Condition – The legend element’s text is now white.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.15.Cleanup” command

### Test Procedure – Configure Hide

**Purpose –** Ensure that the hide configuration works.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.16.Setup” Tcl command
* Pre-Condition – A graph with a single element and the legend are showing.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.16.Body” Tcl command

* Post-Condition – The legend is now hidden.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.16.Cleanup” command

### Test Procedure – Configure Ipadx

**Purpose –** Ensure that the ipadx configuration works.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.17.Setup” Tcl command
* Pre-Condition – A graph with a single element and the legend are showing.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.17.Body” Tcl command

* Post-Condition – The legend’s internal padding on the left and right sides is now 5 pixels wide.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.17.Cleanup” command

### Test Procedure – Configure Ipady

**Purpose –** Ensure that the ipady configuration works.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.18.Setup” Tcl command
* Pre-Condition – A graph with a single element and the legend are showing.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.18.Body” Tcl command

* Post-Condition – The legend’s internal padding on the top and bottom is now 5 pixels wide.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.18.Cleanup” command

### Test Procedure – Configure Padx

**Purpose –** Ensure that the padx configuration works.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.19.Setup” Tcl command
* Pre-Condition – A graph with a single element and the legend are showing.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.19.Body” Tcl command

* Post-Condition – The legend’s external padding on the left and right sides is now 5 pixels wide.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.19.Cleanup” command

### Test Procedure – Configure Pady

**Purpose –** Ensure that the pady configuration works.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.20.Setup” Tcl command
* Pre-Condition – A graph with a single element and the legend are showing.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.20.Body” Tcl command

* Post-Condition – The legend’s internal padding on the top and bottom is now 5 pixels wide.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.20.Cleanup” command

### Test Procedure – Configure Relief Raised

**Purpose –** Ensure that the relief configuration works for raised reliefs.

**Special Requirements –** Borderwidth must be able to be set.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.21.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing. The legend element should also have a border width of 10.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.21.Body” Tcl command

* Post-Condition – The legend element’s relief is now raised.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.21.Cleanup” command

### Test Procedure – Configure Relief Flat

**Purpose –** Ensure that the relief configuration works for flat reliefs.

**Special Requirements –** Borderwidth must be able to be set.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.22.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing. The legend element should also have border width of 10.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.22.Body” Tcl command

* Post-Condition – The legend element’s relief is now flat.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.22.Cleanup” command

### Test Procedure – Configure Relief Grooved

**Purpose –** Ensure that the relief configuration works for grooved reliefs.

**Special Requirements –** Borderwidth must be able to be set.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.23.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing. The legend element should also have an border width of 10.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.23.Body” Tcl command

* Post-Condition – The legend element’s relief is now grooved.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.23.Cleanup” command

### Test Procedure – Configure Relief Ridged

**Purpose –** Ensure that the relief configuration works for ridged reliefs.

**Special Requirements –** Borderwidth must be able to be set.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.24.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing. The legend element should also have an border width of 10.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.24.Body” Tcl command

* Post-Condition – The legend element’s relief is now ridged.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.24.Cleanup” command

### Test Procedure – Configure Relief Solid

**Purpose –** Ensure that the relief configuration works for solid reliefs.

**Special Requirements –** Borderwidth must be able to be set.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.25.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing. The legend element should also have an border width of 10.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.25.Body” Tcl command

* Post-Condition – The legend element’s relief is now solid.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.25.Cleanup” command

### Test Procedure – Configure Relief Sunken

**Purpose –** Ensure that the relief configuration works for sunken reliefs.

**Special Requirements –** Borderwidth must be able to be set.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.26.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing. The legend element should also have an border width of 10.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.26.Body” Tcl command

* Post-Condition – The legend element’s relief is now sunken.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.26.Cleanup” command

### Test Procedure – Configure Shadow

**Purpose –** Ensure that the shadow configuration works for valid colors.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.27.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.27.Body” Tcl command

* Post-Condition – The legend element’s text now has a red shadow.
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.27.Cleanup” command

### Test Procedure – Configure Shadow and Depth

**Purpose –** Ensure that the shadow configuration works for a shadow and a depth.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.8.tcl” file and then call the “graph.legend::RBC.graph.legend.8.28.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing.
* Body

1. Call the “graph.legend::RBC.graph.legend.8.28.Body” Tcl command

* Post-Condition – The legend element’s text now has a red shadow that is 3 pixels behind the text (i.e. it is offset to the bottom right of the text 3 pixels).
* Cleanup – Call the “graph.legend::RBC.graph.legend.8.28.Cleanup” command

### Test Case 9

**Test Case ID –** RBC.graph.legend.9

**Test Item –** The *legend bind* command of the *graph* BLT component.

**Input Specification –** A legend element, a sequence, and a command

**Output Specification –** Command is executed when sequence actions are taken on the legend element.

**Special Procedural Requirements –** None

**Inter-case Dependencies -** RBC.graph.element.create.1

### Test Procedure – Binding Creation

**Purpose –** Ensure that bindings can be created for a legend element.

**Special Requirements –** Legend elements may be activated and deactivated.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.9.tcl” file and then call the “graph.legend::RBC.graph.legend.9.1.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing and it is deactivated.
* Body

1. Call the “graph.legend::RBC.graph.legend.9.1.Body” Tcl command
2. Double-left-click the legend element.

* Post-Condition – The legend element is now activated.
* Cleanup – Call the “graph.legend::RBC.graph.legend.9.1.Cleanup” command

### Test Procedure – Bind Append

**Purpose –** Ensure that bindings can be appended for a legend element.

**Special Requirements –** Legend elements may be activated and deactivated.

**Procedural Steps**

* Setup – Run the “RBC.graph.legend.9.tcl” file and then call the “graph.legend::RBC.graph.legend.9.2.Setup” Tcl command
* Pre-Condition – A graph with a single element is showing and it is deactivated.
* Body

1. Call the “graph.legend::RBC.graph.legend.9.2.Body” Tcl command
2. Double-left-click the legend element.
3. Double-right-click the legend element.

* Post-Condition – The legend element is activated when double-left-clicked and deactivated when double-right-click.
* Cleanup – Call the “graph.legend::RBC.graph.legend.9.2.Cleanup” command

# Graph Line

### Test Cases

**Test Case ID –** RBC.graph.element.\*

**Test Item –** The *line* function of the *graph* BLT component.

**Input Specification –** See element input specification

**Output Specification –** See element output specification

**Special Procedural Requirements –** Replace element component keyword with line

**Inter-case Dependencies –** See element inter-case dependencies

# Graph Marker

### Test Case 1

**Test Case ID –** RBC.graph.marker.1

**Test Item –** The *marker create* command of the *graph* BLT component.

**Input Specification –** The name of the marker and its type

**Output Specification –** The marker element should now exist for the graph

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.create.1

### Test Procedure – Graph Marker Exists

**Purpose –** Ensure creating a marker actually creates the marker on the graph.

**Special Requirements –** None

**TclTest –** RBC.graph.marker.1.1

### Test Procedure – Graph Text Marker Appears

**Purpose –** Ensure creating a text marker actually creates the marker on the graph.

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.1.2.Setup” Tcl command
* Pre-Condition – There is a blank graph on the Tk window
* Body
  1. Run “graph.marker::RBC.graph.marker.1.2.Body” Tcl command
* Post-Condition - A text marker appears on the screen
* Cleanup - Run “graph.marker::RBC.graph.marker.1.2.Cleanup” command

### Test Procedure – Graph Line Marker Appears

**Purpose –** Ensure creating a line marker actually creates the marker on the graph.

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.1.3.Setup” Tcl command
* Pre-Condition – There is a blank graph on the Tk window
* Body

1. Run “graph.marker::RBC.graph.marker.1.3.Body” Tcl command

* Post-Condition - A line marker appears on the screen from the middle to upper right corner
* Cleanup - Run “graph.marker::RBC.graph.marker.1.3.Cleanup” command

### Test Procedure – Graph Bitmap Marker Appears

**Purpose –** Ensure creating a bitmap marker actually creates the marker on the graph.

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.1.4.Setup” Tcl command
* Pre-Condition – There is a blank graph on the Tk window
* Body

1. Run “graph.marker::RBC.graph.marker.1.4.Body” Tcl command

* Post-Condition - A bitmap image appears on the screen
* Cleanup - Run “graph.marker::RBC.graph.marker.1.4.Cleanup” command

### Test Procedure – Graph Image Marker Appears

**Purpose –** Ensure creating a image marker actually creates the marker on the graph.

**Special Requirements –** The greenback.xbm file is in the interpreter directory

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.1.5.Setup” Tcl command
* Pre-Condition – There is a blank graph on the Tk window
* Body

1. Run “graph.marker::RBC.graph.marker.1.5.Body” Tcl command

* Post-Condition – An image appears on the screen
* Cleanup - Run “graph.marker::RBC.graph.marker.1.5.Cleanup” command

### Test Procedure – Graph Polygon Marker Appears

**Purpose –** Ensure creating a polygon marker actually creates the marker on the graph.

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.1.6.Setup” Tcl command
* Pre-Condition – There is a blank graph on the Tk window
* Body

1. Run “graph.marker::RBC.graph.marker.1.6.Body” Tcl command

* Post-Condition - A triangle in the upper right appears on the screen
* Cleanup - Run “graph.marker::RBC.graph.marker.1.6.Cleanup” command

### Test Procedure – Graph Window Marker Appears

**Purpose –** Ensure creating a window marker actually creates the marker on the graph.

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.1.7.Setup” Tcl command
* Pre-Condition – There is a blank graph on the Tk window
* Body

1. Run “graph.marker::RBC.graph.marker.1.7.Body” Tcl command

* Post-Condition – A button appears on the bottom of the graph
* Cleanup - Run “graph.marker::RBC.graph.marker.1.7.Cleanup” command

### Test Case 2

**Test Case ID –** RBC.graph.marker.2

**Test Item –** The *marker destroy* command of the *graph* BLT component.

**Input Specification –** The name of the marker

**Output Specification –** The marker element should be removed from the graph

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.marker.2

### Test Procedure – Deletes a Single Marker

**Purpose –** Ensure the marker delete command works correctly when given a single existing marker name

**Special Requirements –** None

**TclTest –** RBC.graph.marker.2.1

### Test Procedure – Deletes multiple markers

**Purpose –** Ensure the marker delete command works correctly when given multiple existing marker name

**Special Requirements –** None

**TclTest –** RBC.graph.marker.2.2

### Test Procedure – Delete Single Marker Removes from Graph

**Purpose –** Ensure deleting a text marker actually removes the marker on the graph.

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.2.3.Setup” Tcl command
* Pre-Condition – There is a text marker on the graph
* Body

1. Run “graph.marker::RBC.graph.marker.2.3.Body” Tcl command

* Post-Condition – The text marker is gone and a blank graph is showing
* Cleanup - Run “graph.marker::RBC.graph.marker.2.3.Cleanup” command

### Test Procedure – Delete Multiple Marker Removes from Graph

**Purpose –** Ensure deleting multiple markers actually remove them from the graph.

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.2.4.Setup” Tcl command
* Pre-Condition – There is a text marker on the graph
* Body

1. Run “graph.marker::RBC.graph.marker.2.4.Body” Tcl command

* Post-Condition – The text marker is gone and a blank graph is showing
* Cleanup - Run “graph.marker::RBC.graph.marker.2.4.Cleanup” command

### Test Case 3

**Test Case ID –** RBC.graph.marker.3

**Test Item –** The *marker exists* command of the *graph* BLT component.

**Input Specification –** The name of the marker

**Output Specification –** The state of existence of the marker

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.marker.1

### Test Procedure – Non-existant marker

**Purpose –** Ensure that a marker that doesn't exist is recognized by exists

**Special Requirements –** None

**TclTest –** RBC.graph.marker.3.1

### Test Procedure – Existant marker

**Purpose –** Ensure that a marker that an existing marker is recognized by exists

**Special Requirements –** None

**TclTest –** RBC.graph.marker.3.2

### Test Case 4

**Test Case ID –** RBC.graph.marker.4

**Test Item –** The *marker type* command of the *graph* BLT component.

**Input Specification –** The name of the marker

**Output Specification –** The type of the marker

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.marker.4

### Test Procedure – Type of a Text Marker

**Purpose –** Ensure a text marker is of the correct type (TextMarker).

**Special Requirements –** None

**TclTest –** RBC.graph.marker.4.1

### Test Procedure – Type of a Line Marker

**Purpose –** Ensure a line marker is of the correct type (LineMarker).

**Special Requirements –** None

**TclTest –** RBC.graph.marker.4.2

### Test Procedure – Type of a Bitmap Marker

**Purpose –** Ensure a bitmap marker is of the correct type (BitmapMarker).

**Special Requirements –** None

**TclTest –** RBC.graph.marker.4.3

### Test Procedure – Type of a Image Marker

**Purpose –** Ensure a image marker is of the correct type (ImageMarker).

**Special Requirements –** None

**TclTest –** RBC.graph.marker.4.4

### Test Procedure – Type of a Polygon Marker

**Purpose –** Ensure a polygon marker is of the correct type (PolygonMarker).

**Special Requirements –** None

**TclTest –** RBC.graph.marker.4.5

### Test Procedure – Type of a Window Marker

**Purpose –** Ensure a window marker is of the correct type (WindowMarker).

**Special Requirements –** None

**TclTest –** RBC.graph.marker.4.6

### Test Case 5

**Test Case ID –** RBC.graph.marker.5

**Test Item –** The *marker names* command of the *graph* BLT component.

**Input Specification –** A pattern for matching marker names

**Output Specification –** The matching marker names

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.marker.5

### Test Procedure – Marker Names: No Pattern

**Purpose –** Ensure the marker names command works correctly when no pattern is given.

**Special Requirements –** None

**TclTest –** RBC.graph.marker.5.1

### Test Procedure – Marker Names: Exact Pattern

**Purpose –** Ensure the marker names command works correctly when an exact pattern is given.

**Special Requirements –** None

**TclTest –** RBC.graph.marker.5.2

### Test Procedure – Marker Names: Wildcard Pattern

**Purpose –** Ensure the marker names command works correctly when a wildcard pattern is given.

**Special Requirements –** None

**TclTest –** RBC.graph.marker.5.3

### Test Procedure – Marker Names: Incorrect Pattern

**Purpose –** Ensure the marker names command works correctly when an incorrect pattern is given.

**Special Requirements –** None

**TclTest –** RBC.graph.marker.5.4

### Test Case 6

**Test Case ID –** RBC.graph.marker.6

**Test Item –** The *marker configure* command of the *graph* BLT component.

**Input Specification –** A configuration option value pair

**Output Specification –** An updated marker based on configuration

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.marker.6

### Test Procedure – Configure Coords

**Purpose –** Ensure configuring coords changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.6.1

### Test Procedure – Configure Element

**Purpose –** Ensure configuring element changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.6.2

### Test Procedure – Configure Hide

**Purpose –** Ensure configuring hide changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.6.3

### Test Procedure – Configure Under

**Purpose –** Ensure configuring under changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.6.4

### Test Procedure – Configure XOffset

**Purpose –** Ensure configuring xoffset changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.6.5

### Test Procedure – Configure YOffset

**Purpose –** Ensure configuring yoffset changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.6.6

### Test Procedure – Configure Mapx

**Purpose –** Ensure configuring mapx changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.6.7

### Test Procedure – Configure Mapy

**Purpose –** Ensure configuring mapy changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.6.8

### Test Procedure – Displaying Configure Coords

**Purpose –** Ensure coords changes position on screen.

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.6.9.Setup” Tcl command
* Pre-Condition – There is a text marker in the center of the screen
* Body

1. Run “graph.marker::RBC.graph.marker.6.9.Body” Tcl command

* Post-Condition – The text marker is lower than the it was previously

Cleanup - Run “graph.marker::RBC.graph.marker.6.9.Cleanup” command

### Test Procedure – Displaying Configure Element

**Purpose –** Ensure element links to the marker.

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.6.10.Setup” Tcl command
* Pre-Condition – There is a text marker on the graph
* Body

1. Run “graph.marker::RBC.graph.marker.6.10.Body” Tcl command

* Post-Condition – The text marker is gone and a blank graph is showing

Cleanup - Run “graph.marker::RBC.graph.marker.6.10.Cleanup” command

### Test Procedure – Displaying Configure Hide

**Purpose –** Ensure hide removes the marker from view.

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.6.11.Setup” Tcl command
* Pre-Condition – There is a text marker on the graph
* Body

1. Run “graph.marker::RBC.graph.marker.6.11.Body” Tcl command

* Post-Condition – The text marker is gone and a blank graph is showing

Cleanup - Run “graph.marker::RBC.graph.marker.6.11.Cleanup” command

### Test Procedure – Displaying Configure Under

**Purpose –** Ensure under draws the marker below data elements.

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.6.12.Setup” Tcl command
* Pre-Condition – The text marker is drawn above data elements.
* Body

1. Run “graph.marker::RBC.graph.marker.6.12.Body” Tcl command

* Post-Condition – The data elements are now drawn above the marker

Cleanup - Run “graph.marker::RBC.graph.marker.6.12.Cleanup” command

### Test Procedure – Displaying Configure XOffset

**Purpose –** Ensure xoffset draws the marker at a horizontal offset

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.6.13.Setup” Tcl command
* Pre-Condition – There is a text marker in the center of the graph
* Body

1. Run “graph.marker::RBC.graph.marker.6.13.Body” Tcl command

* Post-Condition – The text marker is shifted horizontally to the right

Cleanup - Run “graph.marker::RBC.graph.marker.6.13.Cleanup” command

### Test Procedure – Displaying Configure YOffset

**Purpose –** Ensure yoffset draws the marker at a vertical offset.

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.6.14.Setup” Tcl command
* Pre-Condition – There is a text marker in the center of the graph
* Body

1. Run “graph.marker::RBC.graph.marker.6.14.Body” Tcl command

* Post-Condition – The text marker is shifted down vertically

Cleanup - Run “graph.marker::RBC.graph.marker.6.14.Cleanup” command

### Test Case 7

**Test Case ID –** RBC.graph.marker.7

**Test Item –** The *marker configure* command for text markers of the *graph* BLT component.

**Input Specification –** A configuration option value pair

**Output Specification –** An updated text marker based on configuration

**Special Procedural Requirements –** None

**Inter-case Dependencies –** None

### Test Procedure – Configure Anchor

**Purpose –** Ensure configuring anchor changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.7.1

### Test Procedure – Configure Background

**Purpose –** Ensure configuring background changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.7.2

### Test Procedure – Configure Font

**Purpose –** Ensure configuring font changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.7.3

### Test Procedure – Configure Fill

**Purpose –** Ensure configuring fill changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.7.4

### Test Procedure – Configure Foreground

**Purpose –** Ensure configuring foreground changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.7.5

### Test Procedure – Configure Justify

**Purpose –** Ensure configuring justify changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.7.6

### Test Procedure – Configure Outline

**Purpose –** Ensure configuring outline changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.7.7

### Test Procedure – Configure Padx

**Purpose –** Ensure configuring padx changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.7.8

### Test Procedure – Configure Pady

**Purpose –** Ensure configuring pady changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.7.9

### Test Procedure – Configure Rotate

**Purpose –** Ensure configuring rotate changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.7.10

### Test Procedure – Configure Text

**Purpose –** Ensure configuring text changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.7.11

### Test Procedure – Displaying Configure Anchor

**Purpose –** Ensure anchor changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.7.12.Setup” Tcl command
* Pre-Condition – There is a text marker in the center of the graph
* Body

1. Run “graph.marker::RBC.graph.marker.7.12.Body” Tcl command

* Post-Condition – The text marker is shifted down vertically to its anchor

Cleanup - Run “graph.marker::RBC.graph.marker.7.12.Cleanup” command

### Test Procedure – Displaying Configure Background

**Purpose –** Ensure background changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.7.13.Setup” Tcl command
* Pre-Condition – There is a text marker with a blank background
* Body

1. Run “graph.marker::RBC.graph.marker.7.13.Body” Tcl command

* Post-Condition – The text marker has a red background

Cleanup - Run “graph.marker::RBC.graph.marker.7.13.Cleanup” command

### Test Procedure – Displaying Configure Font

**Purpose –** Ensure font changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.7.14.Setup” Tcl command
* Pre-Condition – There is a text marker has the default font
* Body

1. Run “graph.marker::RBC.graph.marker.7.14.Body” Tcl command

* Post-Condition – The text marker’s font has changed to be huge

Cleanup - Run “graph.marker::RBC.graph.marker.7.14.Cleanup” command

### Test Procedure – Displaying Configure Fill

**Purpose –** Ensure fill changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.7.15.Setup” Tcl command
* Pre-Condition – There is a text marker with a blank background
* Body

1. Run “graph.marker::RBC.graph.marker.7.15.Body” Tcl command

* Post-Condition – The text marker has a red background

Cleanup - Run “graph.marker::RBC.graph.marker.7.15.Cleanup” command

### Test Procedure – Displaying Configure Foreground

**Purpose –** Ensure foreground changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.7.16.Setup” Tcl command
* Pre-Condition – There is a text marker with a blank foreground
* Body

1. Run “graph.marker::RBC.graph.marker.7.16.Body” Tcl command

* Post-Condition – The text marker has a red foreground

Cleanup - Run “graph.marker::RBC.graph.marker.7.16.Cleanup” command

### Test Procedure – Displaying Configure Justify

**Purpose –** Ensure justify changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.7.17.Setup” Tcl command
* Pre-Condition – There is a text marker in the center of the graph
* Body

1. Run “graph.marker::RBC.graph.marker.7.17.Body” Tcl command

* Post-Condition – The text marker’s text now is left justified

Cleanup - Run “graph.marker::RBC.graph.marker.7.17.Cleanup” command

### Test Procedure – Displaying Configure Outline

**Purpose –** Ensure outline changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.7.18.Setup” Tcl command
* Pre-Condition – There is a text marker with a blank foreground
* Body

1. Run “graph.marker::RBC.graph.marker.7.18.Body” Tcl command

* Post-Condition – The text marker has a red foreground

Cleanup - Run “graph.marker::RBC.graph.marker.7.18.Cleanup” command

### Test Procedure – Displaying Configure Padx

**Purpose –** Ensure padx changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.7.19.Setup” Tcl command
* Pre-Condition – There is a text marker with an even horizontal padding
* Body

1. Run “graph.marker::RBC.graph.marker.7.19.Body” Tcl command

* Post-Condition – The text marker’s horizontal padding is increased and uneven

Cleanup - Run “graph.marker::RBC.graph.marker.7.19.Cleanup” command

### Test Procedure – Displaying Configure Pady

**Purpose –** Ensure pady changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.7.20.Setup” Tcl command
* Pre-Condition – There is a text marker with an even vertical padding
* Body

1. Run “graph.marker::RBC.graph.marker.7.20.Body” Tcl command

* Post-Condition – The text marker’s vertical padding is increased and uneven

Cleanup - Run “graph.marker::RBC.graph.marker.7.20.Cleanup” command

### Test Procedure – Displaying Configure Rotate

**Purpose –** Ensure rotate changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.7.21.Setup” Tcl command
* Pre-Condition – There is a text marker says ‘Marker 1’
* Body

1. Run “graph.marker::RBC.graph.marker.7.21.Body” Tcl command

* Post-Condition – The text marker says ‘Text’

Cleanup - Run “graph.marker::RBC.graph.marker.7.21.Cleanup” command

### Test Procedure – Displaying Configure Text

**Purpose –** Ensure text changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.7.22.Setup” Tcl command
* Pre-Condition – There is a text marker in the center of the graph
* Body

1. Run “graph.marker::RBC.graph.marker.7.22.Body” Tcl command

* Post-Condition – The text marker is shifted down vertically

Cleanup - Run “graph.marker::RBC.graph.marker.7.22.Cleanup” command

### Test Procedure – Displaying Configure Shadow

**Purpose –** Ensure Shadow changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.7.23.Setup” Tcl command
* Pre-Condition – There is a text marker in the center of the graph
* Body

1. Run “graph.marker::RBC.graph.marker.7.23.Body” Tcl command

* Post-Condition – The text marker has a red shadow behind it

Cleanup - Run “graph.marker::RBC.graph.marker.7.23.Cleanup” command

### Test Case 8

**Test Case ID –** RBC.graph.marker.8

**Test Item –** The *marker configure* command for line markers of the *graph* BLT component.

**Input Specification –** A configuration option value pair

**Output Specification –** An updated line marker based on configuration

**Special Procedural Requirements –** None

**Inter-case Dependencies –** None

### Test Procedure – Configure Dashes

**Purpose –** Ensure configuring dashes changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.8.1

### Test Procedure – Configure Fill

**Purpose –** Ensure configuring fill changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.8.2

### Test Procedure – Configure Linewidth

**Purpose –** Ensure configuring linewidth changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.8.3

### Test Procedure – Configure Outline

**Purpose –** Ensure configuring outline changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.8.4

### Test Procedure – Configure Xor

**Purpose –** Ensure configuring xor changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.8.5

### Test Procedure – Displaying Configure Dashes

**Purpose –** Ensure dashes changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.8.6.Setup” Tcl command
* Pre-Condition – There is a line marker in the center of the graph
* Body

1. Run “graph.marker::RBC.graph.marker.8.6.Body” Tcl command

* Post-Condition – The line marker changes to a dashed pattern

Cleanup - Run “graph.marker::RBC.graph.marker.8.6.Cleanup” command

### Test Procedure – Displaying Configure Fill (No effect)

**Purpose –** Ensure fill changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.8.7.Setup” Tcl command
* Pre-Condition – There is a line marker in the center of the graph
* Body

1. Run “graph.marker::RBC.graph.marker.8.7.Body” Tcl command

* Post-Condition – No effect

Cleanup - Run “graph.marker::RBC.graph.marker.8.7.Cleanup” command

### Test Procedure – Displaying Configure Linewidth

**Purpose –** Ensure linewidth changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.8.8.Setup” Tcl command
* Pre-Condition – There is a line marker in the center of the graph
* Body

1. Run “graph.marker::RBC.graph.marker.8.8.Body” Tcl command

* Post-Condition – The line marker changes to be thicker

Cleanup - Run “graph.marker::RBC.graph.marker.8.8.Cleanup” command

### Test Procedure – Displaying Configure Outline

**Purpose –** Ensure outline changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.8.9.Setup” Tcl command
* Pre-Condition – There is a line marker in the center of the graph
* Body

1. Run “graph.marker::RBC.graph.marker.8.9.Body” Tcl command

* Post-Condition – The line changes color to red

Cleanup - Run “graph.marker::RBC.graph.marker.8.9.Cleanup” command

### Test Procedure – Displaying Configure Outline

**Purpose –** Ensure outline changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.8.9.Setup” Tcl command
* Pre-Condition – There is a line marker in the center of the graph
* Body

1. Run “graph.marker::RBC.graph.marker.8.9.Body” Tcl command

* Post-Condition – The line changes color to red

Cleanup - Run “graph.marker::RBC.graph.marker.8.9.Cleanup” command

### Test Case 9

**Test Case ID –** RBC.graph.marker.9

**Test Item –** The *marker configure* command for bitmap markers of the *graph* BLT component.

**Input Specification –** A configuration option value pair

**Output Specification –** An updated bitmap marker based on configuration

**Special Procedural Requirements –** The greenback.xbm file needs to be in the executing directory

**Inter-case Dependencies –** None

### Test Procedure – Configure Background

**Purpose –** Ensure configuring background changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.9.1

### Test Procedure – Configure Bitmap

**Purpose –** Ensure configuring bitmap changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.9.2

### Test Procedure – Configure Fill

**Purpose –** Ensure configuring fill changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.9.3

### Test Procedure – Configure Foreground

**Purpose –** Ensure configuring foreground changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.9.4

### Test Procedure – Configure Outline

**Purpose –** Ensure configuring outline changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.9.5

### Test Procedure – Configure Rotate

**Purpose –** Ensure configuring rotate changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.9.6

### Test Procedure – Displaying Configure Background

**Purpose –** Ensure background changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.9.7.Setup” Tcl command
* Pre-Condition – A bitmap of a dollar bill is in the lower left of a graph
* Body

1. Run “graph.marker::RBC.graph.marker.9.7.Body” Tcl command

* Post-Condition – The back of the bitmap changes color to red

Cleanup - Run “graph.marker::RBC.graph.marker.9.7.Cleanup” command

### Test Procedure – Displaying Configure Bitmap

**Purpose –** Ensure bitmap changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.9.8.Setup” Tcl command
* Pre-Condition – A blank graph window appears
* Body

1. Run “graph.marker::RBC.graph.marker.9.8.Body” Tcl command

* Post-Condition – A bitmap of a dollar bill is in the lower left of a graph

Cleanup - Run “graph.marker::RBC.graph.marker.9.8.Cleanup” command

### Test Procedure – Displaying Configure Fill

**Purpose –** Ensure fill changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.9.9.Setup” Tcl command
* Pre-Condition – A bitmap of a dollar bill is in the lower left of a graph
* Body

1. Run “graph.marker::RBC.graph.marker.9.9.Body” Tcl command

* Post-Condition – The back of the bitmap changes color to red

Cleanup - Run “graph.marker::RBC.graph.marker.9.9.Cleanup” command

### Test Procedure – Displaying Configure Foreground

**Purpose –** Ensure foreground changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.9.10.Setup” Tcl command
* Pre-Condition – A bitmap of a dollar bill is in the lower left of a graph
* Body

1. Run “graph.marker::RBC.graph.marker.9.10.Body” Tcl command

* Post-Condition – The front writing of the bitmap changes color to red

Cleanup - Run “graph.marker::RBC.graph.marker.9.10.Cleanup” command

### Test Procedure – Displaying Configure Outline

**Purpose –** Ensure outline changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.9.11.Setup” Tcl command
* Pre-Condition – A bitmap of a dollar bill is in the lower left of a graph
* Body

1. Run “graph.marker::RBC.graph.marker.9.11.Body” Tcl command

* Post-Condition – The front writing of the bitmap changes color to red

Cleanup - Run “graph.marker::RBC.graph.marker.9.11.Cleanup” command

### Test Procedure – Displaying Configure Rotate

**Purpose –** Ensure rotate changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.9.12.Setup” Tcl command
* Pre-Condition – A bitmap of a dollar bill is in the lower left of a graph
* Body

1. Run “graph.marker::RBC.graph.marker.9.12.Body” Tcl command

* Post-Condition – The bitmap rotates so the right side is facing towards the top right

Cleanup - Run “graph.marker::RBC.graph.marker.9.12.Cleanup” command

### Test Procedure – Displaying Configure Anchor

**Purpose –** Ensure anchor changes on screen

**Special Requirements –** none

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.9.13.Setup” Tcl command
* Pre-Condition – A bitmap of a dollar bill is in the lower left of a graph
* Body

1. Run “graph.marker::RBC.graph.marker.9.13.Body” Tcl command

* Post-Condition – The bitmap lowers from its previous position

Cleanup - Run “graph.marker::RBC.graph.marker.9.13.Cleanup” command

### Test Case 10

**Test Case ID –** RBC.graph.marker.10

**Test Item –** The *marker configure* command for image markers of the *graph* BLT component.

**Input Specification –** A configuration option value pair

**Output Specification –** An updated image marker based on configuration

**Special Procedural Requirements –** None

**Inter-case Dependencies –** None

### Test Procedure – Configure Anchor

**Purpose –** Ensure configuring Anchor changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.10.1

### Test Procedure – Configure Image

**Purpose –** Ensure configuring image changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.10.2

### Test Procedure – Displaying Configure Anchor

**Purpose –** Ensure anchor changes on screen

**Special Requirements –** buckskin.gif needs to be in executing directory

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.10.3.Setup” Tcl command
* Pre-Condition – An image of a texture is in the lower right of a graph
* Body

1. Run “graph.marker::RBC.graph.marker.10.3.Body” Tcl command

* Post-Condition – The image lowers from its previous position

Cleanup - Run “graph.marker::RBC.graph.marker.10.3.Cleanup” command

### Test Procedure – Displaying Configure Image

**Purpose –** Ensure image changes on screen

**Special Requirements –** buckskin.gif and stopsign.gif need to be in the executing directory

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.10.4.Setup” Tcl command
* Pre-Condition – An image of a texture is in the lower right of a graph
* Body

1. Run “graph.marker::RBC.graph.marker.10.4.Body” Tcl command

* Post-Condition – The image changes to one of a stop sign

Cleanup - Run “graph.marker::RBC.graph.marker.10.4.Cleanup” command

### Test Case 11

**Test Case ID –** RBC.graph.marker.11

**Test Item –** The *marker configure* command for polygon markers of the *graph* BLT component.

**Input Specification –** A configuration option value pair

**Output Specification –** An updated polygon marker based on configuration

**Special Procedural Requirements –** None

**Inter-case Dependencies –** None

### Test Procedure – Configure Dashes

**Purpose –** Ensure configuring dashes changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.11.1

### Test Procedure – Configure Fill

**Purpose –** Ensure configuring fill changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.11.2

### Test Procedure – Configure Linewidth

**Purpose –** Ensure configuring linewidth changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.11.3

### Test Procedure – Configure Outline

**Purpose –** Ensure configuring outline changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.11.4

### Test Procedure – Configure Stipple

**Purpose –** Ensure configuring stipple changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.11.5

### Test Procedure – Displaying Configure Dashes

**Purpose –** Ensure dashes changes on screen

**Special Requirements –** None

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.11.6.Setup” Tcl command
* Pre-Condition – A triangle is drawn across the upper left of a graph
* Body

1. Run “graph.marker::RBC.graph.marker.11.6.Body” Tcl command

* Post-Condition – The triangle lines have a dashed patterns

Cleanup - Run “graph.marker::RBC.graph.marker.11.6.Cleanup” command

### Test Procedure – Displaying Configure Fill

**Purpose –** Ensure fill changes on screen

**Special Requirements –** None

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.11.7.Setup” Tcl command
* Pre-Condition – A triangle is drawn across the upper left of a graph
* Body

1. Run “graph.marker::RBC.graph.marker.11.7.Body” Tcl command

* Post-Condition – The triangle is filled with blue

Cleanup - Run “graph.marker::RBC.graph.marker.11.7.Cleanup” command

### Test Procedure – Displaying Configure Linewidth

**Purpose –** Ensure linewdith changes on screen

**Special Requirements –** None

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.11.8.Setup” Tcl command
* Pre-Condition – A triangle is drawn across the upper left of a graph
* Body

1. Run “graph.marker::RBC.graph.marker.11.8.Body” Tcl command

* Post-Condition – The triangle lines increase in width

Cleanup - Run “graph.marker::RBC.graph.marker.11.8.Cleanup” command

### Test Procedure – Displaying Configure Outline

**Purpose –** Ensure outline changes on screen

**Special Requirements –** None

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.11.9.Setup” Tcl command
* Pre-Condition – A triangle is drawn across the upper left of a graph
* Body

1. Run “graph.marker::RBC.graph.marker.11.9.Body” Tcl command

* Post-Condition – The triangle lines are changed to a green color

Cleanup - Run “graph.marker::RBC.graph.marker.11.9.Cleanup” command

### Test Procedure – Displaying Configure Stipple

**Purpose –** Ensure stipple changes on screen

**Special Requirements –** greenback.xbm is in the executing directory

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.11.10.Setup” Tcl command
* Pre-Condition – A triangle is drawn across the upper left of a graph
* Body

1. Run “graph.marker::RBC.graph.marker.11.10.Body” Tcl command

* Post-Condition – The triangle has a fill of a dollar bill image.

Cleanup - Run “graph.marker::RBC.graph.marker.11.10.Cleanup” command

### Test Case 12

**Test Case ID –** RBC.graph.marker.12

**Test Item –** The *marker configure* command for window markers of the *graph* BLT component.

**Input Specification –** A configuration option value pair

**Output Specification –** An updated window marker based on configuration

**Special Procedural Requirements –** None

**Inter-case Dependencies –** None

### Test Procedure – Configure Anchor

**Purpose –** Ensure configuring anchor changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.12.1

### Test Procedure – Configure Height

**Purpose –** Ensure configuring height changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.12.2

### Test Procedure – Configure Width

**Purpose –** Ensure configuring width changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.12.3

### Test Procedure – Configure Window

**Purpose –** Ensure configuring window changes the corresponding option value pair

**Special Requirements –** None

**TclTest –** RBC.graph.marker.12.4

### Test Procedure – Displaying Configure Anchor

**Purpose –** Ensure anchor changes on screen

**Special Requirements –** None

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.12.5.Setup” Tcl command
* Pre-Condition – A button with Button text is in the lower right of the graph
* Body

1. Run “graph.marker::RBC.graph.marker.12.5.Body” Tcl command

* Post-Condition – The button is lowered from its previous position

Cleanup - Run “graph.marker::RBC.graph.marker.12.5.Cleanup” command

### Test Procedure – Displaying Configure Height

**Purpose –** Ensure height changes on screen

**Special Requirements –** None

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.12.6.Setup” Tcl command
* Pre-Condition – A button with Button text is in the lower right of the graph
* Body

1. Run “graph.marker::RBC.graph.marker.12.6.Body” Tcl command

* Post-Condition – The button shrinks in height

Cleanup - Run “graph.marker::RBC.graph.marker.12.6.Cleanup” command

### Test Procedure – Displaying Configure Width

**Purpose –** Ensure width changes on screen

**Special Requirements –** None

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.12.7.Setup” Tcl command
* Pre-Condition – A button with Button text is in the lower right of the graph
* Body

1. Run “graph.marker::RBC.graph.marker.12.7.Body” Tcl command

* Post-Condition – The button shrinks in width

Cleanup - Run “graph.marker::RBC.graph.marker.12.7.Cleanup” command

### Test Procedure – Displaying Configure Window

**Purpose –** Ensure window changes on screen

**Special Requirements –** None

**Procedural Steps**

* Setup –Run the “graph.marker::RBC.graph.marker.12.8.Setup” Tcl command
* Pre-Condition – A button with Button text is in the lower right of the graph
* Body

1. Run “graph.marker::RBC.graph.marker.12.8.Body” Tcl command

* Post-Condition – The button changes to a new button with text “test”.

Cleanup - Run “graph.marker::RBC.graph.marker.12.8.Cleanup” command

### Test Case 13

**Test Case ID –** RBC.graph.marker.13

**Test Item –** The *marker bind* command of the *graph* BLT component.

**Input Specification –** A marker or arbitrary tag name, an action sequence, and a command

**Output Specification –** *marker bind* should return the bound actions and commands

**Special Procedural Requirements –** None

**Inter-case Dependencies -** RBC.graph.marker.create.1

### Test Procedure – Bind Create

**Purpose –** Ensure that bindings can be created for a marker

**Special Requirements –** None

**TclTest –** RBC.graph.marker.bind.1.1

### Test Procedure – Bind Query for Sequence and Marker

**Purpose –** Ensure that bindings can be queried for a sequence and marker

**Special Requirements –** None

**TclTest –** RBC.graph.marker.bind.1.2

### Test Procedure – Bind Command Append

**Purpose –** Ensure that bindings can be appended for a sequence and marker

**Special Requirements –** None

**TclTest –** RBC.graph.marker.bind.1.3

### Test Procedure – Bind Command Overwrite

**Purpose –** Ensure that bindings are overwritten for a sequence and marker

**Special Requirements –** None

**TclTest –** RBC.graph.marker.bind.1.4

### Test Procedure – Bind Query for Element

**Purpose –** Ensure that bound sequences can be queried for just a marker

**Special Requirements –** None

**TclTest –** RBC.graph.marker.bind.1.5

### Test Procedure – Bind actions execute

**Purpose –** Ensure bind actions execute

**Special Requirements –** None

**Procedural Steps**

* Setup –Run the “graph.marker:: RBC.graph.marker.13.6.Setup” Tcl command
* Pre-Condition – A Text marker appears in the middle of a plot
* Body

1. Run “graph.marker::RBC.graph.marker.13.6.Body” Tcl command
2. Double click the text marker

* Post-Condition – The double click should output onto console the word “test”

Cleanup - Run “graph.marker::RBC.graph.marker.13.6.Cleanup” command

# Graph Pen

### Test Case 1

**Test Case ID –** RBC.graph.pen.1

**Test Item –** The *pen create* command of the *graph* BLT component.

**Input Specification –** A *name* and any *option*-*value* pairs

**Output Specification –** A pen with *name* and configuration that reflects the *option*-*value* pairs exists

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.pen.4, RBC.graph.pen.5

### Test Procedure – Create

**Purpose –** Ensure that pen creation works when passed only a pen name.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.1.1

### Test Procedure – Create Non-overwriting

**Purpose –** Ensure that pen creation will not overwrite an existing pen name.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.1.2

### Test Procedure – Default Pens

**Purpose –** Ensure that two default pens are created automatically when a graph is created

**Special Requirements –** None

**TclTest –** RBC.graph.pen.1.3

### Test Procedure – Creation with Single Option

**Purpose –** Ensure that pen creation works when passed a single option-value pair.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.1.4

### Test Procedure – Creation with Multiple Options

**Purpose –** Ensure that pen creation works when passed option-value pairs.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.1.5

### Test Case 2

**Test Case ID –** RBC.graph.pen.2

**Test Item –** The *pen delete* command of the *graph* BLT component.

**Input Specification –** An existing pen

**Output Specification –** The pen should no longer exist

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.pen.1, RBC.graph.pen.4

### Test Procedure – Deleting Single Pen

**Purpose –** Ensure that pen deletion works when deleting a single pen.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.2.1

### Test Procedure – Deleting Multiple Pens

**Purpose –** Ensure that pen deletion works when deleting multiple pens.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.2.2

### Test Case 3

**Test Case ID –** RBC.graph.pen.3

**Test Item –** The *pen configure* command of the *graph* BLT component.

**Input Specification –** A valid configuration *option* flag and *value* pair

**Output Specification –** *pen cget* *-option* should return *value*

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.pen.1, RBC.graph.pen.5

### Test Procedure – Configure Color

**Purpose –** Ensure that the color configuration works for valid colors.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.1

### Test Procedure – Configure Dashes

**Purpose –** Ensure that the dashes configuration works for valid dashes.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.2

### Test Procedure – Configure No Dashes

**Purpose –** Ensure that the dashes configuration works for no dashes.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.3

### Test Procedure – Configure Fill Color

**Purpose –** Ensure that the fill configuration works for valid fill colors.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.4

### Test Procedure – Configure No Fill

**Purpose –** Ensure that the fill configuration works for no fill.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.5

### Test Procedure – Configure Line Width

**Purpose –** Ensure that the linewidth configuration works for valid linewidths.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.6

### Test Procedure – Configure Off Dash Color

**Purpose –** Ensure that the offdash configuration works for valid offdash colors.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.7

### Test Procedure – Configure Outline Color

**Purpose –** Ensure that the outline configuration works for valid outline colors.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.8

### Test Procedure – Configure Outline Default Color

**Purpose –** Ensure that the outline configuration works for defcolor (the same color as the color configuration option).

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.9

### Test Procedure – Configure Outline Width

**Purpose –** Ensure that the outlinewidth configuration works for valid widths.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.10

### Test Procedure – Configure No Outline

**Purpose –** Ensure that the outlinewidth configuration works for no width.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.11

### Test Procedure – Configure Symbol Size

**Purpose –** Ensure that the pixels configuration works for valid symbol sizes.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.12

### Test Procedure – Configure No Symbol (Size Zero)

**Purpose –** Ensure that the pixels configuration works for symbole size zero.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.13

### Test Procedure – Configure Square Symbol

**Purpose –** Ensure that the symbol configuration works for square.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.14

### Test Procedure – Configure Circle Symbol

**Purpose –** Ensure that the symbol configuration works for circle.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.15

### Test Procedure – Configure Diamond Symbol

**Purpose –** Ensure that the symbol configuration works for diamond.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.16

### Test Procedure – Configure Plus Symbol

**Purpose –** Ensure that the symbol configuration works for plus.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.17

### Test Procedure – Configure Cross Symbol

**Purpose –** Ensure that the symbol configuration works for cross.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.18

### Test Procedure – Configure Splus Symbol

**Purpose –** Ensure that the symbol configuration works for splus.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.19

### Test Procedure – Configure Scross Symbol

**Purpose –** Ensure that the symbol configuration works for scross.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.20

### Test Procedure – Configure Triangle Symbol

**Purpose –** Ensure that the symbol configuration works for triangle.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.21

### Test Procedure – Configure No Symbol

**Purpose –** Ensure that the symbol configuration works for no symbol.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.22

### Test Procedure – Configure Bitmap Symbol

**Purpose –** Ensure that the symbol configuration works for bitmap symbols.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.23

### Test Procedure – Configure Default Type

**Purpose –** Ensure that the default type of a pen is set.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.24

### Test Procedure – Configure Type

**Purpose –** Ensure that the type configuration works.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.3.25

### Test Case 4

**Test Case ID –** RBC.graph.pen.4

**Test Item –** The *pen names* command of the *graph* BLT component.

**Input Specification –** An optional *pattern*

**Output Specification –** A list of all the names of the current pens that match *pattern*

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.pen.1

### Test Procedure – Names No Pattern

**Purpose –** Ensure that names works with no given pattern.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.4.1

### Test Procedure – Names with Pattern

**Purpose –** Ensure that names works with a pattern parameter.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.4.2

### Test Case 5

**Test Case ID –** RBC.graph.pen.5

**Test Item –** The *pen cget* command of the *graph* BLT component.

**Input Specification –** A configuration *option* flag

**Output Specification –** The current value for the *option* flag

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.pen.1, RBC.graph.pen.3

### Test Procedure – Cget

**Purpose –** Ensure that cget works with an explicitly set option.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.5.1

### Test Procedure – Cget Default

**Purpose –** Ensure that cget works with a default value.

**Special Requirements –** None

**TclTest –** RBC.graph.pen.5.2

### Test Case 6

**Test Case ID –** RBC.graph.pen.6

**Test Item –** The *pen configure* command of the *graph* BLT component.

**Input Specification –** A configuration *option* flag

**Output Specification –** Elements on the graph using the pen update to reflect the new configuration values

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.pen.1, RBC.graph.element.2, RBC.graph.element.3

### Test Procedure – Configure Color

**Purpose –** Ensure that the color configuration works for valid colors.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.pen.6.tcl” file and then call the “graph.pen::RBC.graph.pen.6.1.Setup” Tcl command
* Pre-Condition – A graph with two blue points and a single blue line is showing.
* Body

1. Call the “graph.pen::RBC.graph.pen.6.1.Body” Tcl command

* Post-Condition – The previously displaying line is now red. This includes the line, the symbols, their outlines and fills, as well as the legend entry.
* Cleanup – Call the “graph.pen::RBC.graph.pen.6.1.Cleanup” command

### Test Procedure – Configure Dashes

**Purpose –** Ensure that the dashes configuration works for valid dashes.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.pen.6.tcl” file and then call the “graph.pen::RBC.graph.pen.6.2.Setup” Tcl command
* Pre-Condition – A graph with two blue points and a single blue line is showing.
* Body

1. Call the “graph.pen::RBC.graph.pen.6.2.Body” Tcl command

* Post-Condition – The previously displaying line is now dashed. The dashes are 10 pixels long, while the gaps in between them are only 3 pixels long.
* Cleanup – Call the “graph.pen::RBC.graph.pen.6.2.Cleanup” command

### Test Procedure – Configure No Dashes

**Purpose –** Ensure that the dashes configuration works for no dashes.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.pen.6.tcl” file and then call the “graph.pen::RBC.graph.pen.6.3.Setup” Tcl command
* Pre-Condition – A graph with two blue points and a single dashed blue line is showing.
* Body

1. Call the “graph.pen::RBC.graph.pen.6.3.Body” Tcl command

* Post-Condition – The previously displaying dashed line is now solid.
* Cleanup – Call the “graph.pen::RBC.graph.pen.6.3.Cleanup” command

### Test Procedure – Configure Fill

**Purpose –** Ensure that the fill configuration works for valid fill colors.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.pen.6.tcl” file and then call the “graph.pen::RBC.graph.pen.6.4.Setup” Tcl command
* Pre-Condition – A graph with two blue points and a single blue line is showing.
* Body

1. Call the “graph.pen::RBC.graph.pen.6.4.Body” Tcl command

* Post-Condition – The previously displaying points are now yellow with a blue outline. The legend also updates to reflect this change.
* Cleanup – Call the “graph.pen::RBC.graph.pen.6.4.Cleanup” command

### Test Procedure – Configure No Fill

**Purpose –** Ensure that the fill configuration works for no fill.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.pen.6.tcl” file and then call the “graph.pen::RBC.graph.pen.6.5.Setup” Tcl command
* Pre-Condition – A graph with two blue points and a single blue line is showing.
* Body

1. Call the “graph.pen::RBC.graph.pen.6.5.Body” Tcl command

* Post-Condition – The previously displaying points are no longer filled (i.e. their fill appears to be the background color) with a blue outline. The legend also updates to reflect this change.
* Cleanup – Call the “graph.pen::RBC.graph.pen.6.5.Cleanup” command

### Test Procedure – Configure Line Width

**Purpose –** Ensure that the linewidth configuration works for valid linewidths.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.pen.6.tcl” file and then call the “graph.pen::RBC.graph.pen.6.6.Setup” Tcl command
* Pre-Condition – A graph with two blue points and a single blue line is showing.
* Body

1. Call the “graph.pen::RBC.graph.pen.6.6.Body” Tcl command

* Post-Condition – The previously displaying line is now 10 pixels thick, as is the line displaying in the legend.
* Cleanup – Call the “graph.pen::RBC.graph.pen.6.6.Cleanup” command

### Test Procedure – Configure Offdash Color

**Purpose –** Ensure that the offdash configuration works for valid offdash colors.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.pen.6.tcl” file and then call the “graph.pen::RBC.graph.pen.6.7.Setup” Tcl command
* Pre-Condition – A graph with two blue points and a single, dashed blue line is showing.
* Body

1. Call the “graph.pen::RBC.graph.pen.6.7.Body” Tcl command

* Post-Condition – The gaps in the line that were previously clear (i.e. the same color as the background) are now red.
* Cleanup – Call the “graph.pen::RBC.graph.pen.6.7.Cleanup” command

### Test Procedure – Configure Outline Color

**Purpose –** Ensure that the outline configuration works for valid outline colors.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.pen.6.tcl” file and then call the “graph.pen::RBC.graph.pen.6.8.Setup” Tcl command
* Pre-Condition – A graph with two blue points and a single blue line is showing.
* Body

1. Call the “graph.pen::RBC.graph.pen.6.8.Body” Tcl command

* Post-Condition – The two blue points are now outlined in red.
* Cleanup – Call the “graph.pen::RBC.graph.pen.6.8.Cleanup” command

### Test Procedure – Configure Outline Color to Default Color

**Purpose –** Ensure that the outline configuration works for defcolor (the same color as the *color* configuration option).

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.pen.6.tcl” file and then call the “graph.pen::RBC.graph.pen.6.9.Setup” Tcl command
* Pre-Condition – A graph with two blue points and a single blue line is showing.
* Body

1. Call the “graph.pen::RBC.graph.pen.6.9.Body” Tcl command

* Post-Condition – The two blue points, their outlines and fills, as well as the line between the points are now red.
* Cleanup – Call the “graph.pen::RBC.graph.pen.6.9.Cleanup” command

### Test Procedure – Configure Outline Width

**Purpose –** Ensure that the outlinewidth configuration works for valid widths.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.pen.6.tcl” file and then call the “graph.pen::RBC.graph.pen.6.10.Setup” Tcl command
* Pre-Condition – A graph with two blue points outlined in red and a single blue line is showing.
* Body

1. Call the “graph.pen::RBC.graph.pen.6.10.Body” Tcl command

* Post-Condition – The red outlines of the points are now 5 pixels thick.
* Cleanup – Call the “graph.pen::RBC.graph.pen.6.10.Cleanup” command

### Test Procedure – Configure Outline Width of Zero

**Purpose –** Ensure that the outlinewidth configuration works for zero width.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.pen.6.tcl” file and then call the “graph.pen::RBC.graph.pen.6.11.Setup” Tcl command
* Pre-Condition – A graph with two blue points outlined in red and a single blue line is showing.
* Body

1. Call the “graph.pen::RBC.graph.pen.6.11.Body” Tcl command

* Post-Condition – The two blue points no longer have an outline.
* Cleanup – Call the “graph.pen::RBC.graph.pen.6.11.Cleanup” command

### Test Procedure – Configure Symbol Size

**Purpose –** Ensure that the pixels configuration works for valid symbol sizes.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.pen.6.tcl” file and then call the “graph.pen::RBC.graph.pen.6.12.Setup” Tcl command
* Pre-Condition – A graph with two blue points and a single blue line is showing.
* Body

1. Call the “graph.pen::RBC.graph.pen.6.12.Body” Tcl command
2. Resize the wish84 slightly by dragging the corner of the window

* Post-Condition – The two blue points are now 30 pixels in diameter.
* Cleanup – Call the “graph.pen::RBC.graph.pen.6.12.Cleanup” command

### Test Procedure – Configure Symbol Size Zero

**Purpose –** Ensure that the pixels configuration works for symbol size zero.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.pen.6.tcl” file and then call the “graph.pen::RBC.graph.pen.6.13.Setup” Tcl command
* Pre-Condition – A graph with two blue points and a single blue line is showing.
* Body

1. Call the “graph.pen::RBC.graph.pen.6.13.Body” Tcl command
2. Resize the wish84 slightly by dragging the corner of the window

* Post-Condition – The two blue points are now 0 pixels in diameter (i.e. not visible)
* Cleanup – Call the “graph.pen::RBC.graph.pen.6.13.Cleanup” command

### Test Procedure – Configure Symbols

**Purpose –** Ensure that the symbol configuration works for valid symbols.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.pen.6.tcl” file and then call the “graph.pen::RBC.graph.pen.6.14.Setup” Tcl command
* Pre-Condition – A graph with 10 lines, each with 3 points, is now showing. The points should all be circles.
* Body

1. Call the “graph.pen::RBC.graph.pen.6.14.Body” Tcl command

* Post-Condition – The 10 lines, from steepest to shallowest should have the following symbols:
  1. An exclamation point
  2. Nothing
  3. A triangle
  4. An x-shaped symbol whose lines are 1 pixel thick
  5. A +-shaped symbol whose lines are 1 pixel thick
  6. An x-shaped symbol with thick lines
  7. A +-shaped symbol with thick lines
  8. A diamond
  9. A circle
  10. A square
* Cleanup – Call the “graph.pen::RBC.graph.pen.6.14.Cleanup” command

### Test Case 7

**Test Case ID –** RBC.graph.pen.7

**Test Item –** The *pen delete* command of the *graph* BLT component.

**Input Specification –** A pen

**Output Specification –** The pen is deleted and all elements using the pen still reflect the pen’s configuration (i.e. deleting a pen does not affect any elements using the pen)

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.pen.1, RBC.graph.element.2, RBC.graph.element.3

### Test Procedure – Pen Deletion

**Purpose –** Ensure deleting a pen does not affect elements using the pen.

**Special Requirements –** None

**Procedural Steps**

* Setup – Run the “RBC.graph.pen.7.tcl” file and then call the “graph.pen::RBC.graph.pen.7.1.Setup” Tcl command
* Pre-Condition – A graph with two red points and a single red line is showing.
* Body

1. Call the “graph.pen::RBC.graph.pen.7.1.Body” Tcl command

* Post-Condition – A graph with two red points and a single red line is showing.
* Cleanup – Call the “graph.pen::RBC.graph.pen.7.1.Cleanup” command

# Graph PostScript

### Test Case 1

**Test Case ID –** RBC.graph.postscript.1

**Test Item –** The *postscript cget* command of the *graph* BLT component.

**Input Specification –** A configuration *option* flag

**Output Specification –** The current value for the *option* flag

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.postscript.2

### Test Procedure – Cget

**Purpose –** Ensure that cget works for default values.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.1.1

### Test Procedure – Cget Default

**Purpose –** Ensure that cget works with an explicitly set option.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.1.2

### Test Case 2

**Test Case ID –** RBC.graph.postscript.2

**Test Item –** The *postscript configure* command of the *graph* BLT component.

**Input Specification –** A valid configuration *option* flag and *value* pair

**Output Specification –** *postscript cget -option* should return *value*

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.graph.postscript.1

### Test Procedure – Configure Center False

**Purpose –** Ensure that the center configuration works for valid booleans.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.1

### Test Procedure – Configure Center True

**Purpose –** Ensure that the center configuration works for valid booleans.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.2

### Test Procedure – Configure Colormode Color

**Purpose –** Ensure that the colormode configuration works for color mode.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.3

### Test Procedure – Configure Colormode Gray

**Purpose –** Ensure that the colormode configuration works for gray mode.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.4

### Test Procedure – Configure Colormode Greyscale

**Purpose –** Ensure that the colormode configuration works for grayscale mode.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.5

### Test Procedure – Configure Colormode Mono

**Purpose –** Ensure that the colormode configuration works for mono mode.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.6

### Test Procedure – Configure Colormode Monochrome

**Purpose –** Ensure that the colormode configuration works for monochrome mode.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.7

### Test Procedure – Configure Decorations False

**Purpose –** Ensure that the decorations configuration works.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.8

### Test Procedure – Configure Decorations True

**Purpose –** Ensure that the decorations configuration works.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.9

### Test Procedure – Configure Height Zero

**Purpose –** Ensure that the height configuration works for zero.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.10

### Test Procedure – Configure Height

**Purpose –** Ensure that the height configuration works for valid values.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.11

### Test Procedure – Configure Landscape False

**Purpose –** Ensure that the landscape configuration works.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.12

### Test Procedure – Configure Landscape True

**Purpose –** Ensure that the landscape configuration works.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.13

### Test Procedure – Configure Maxpect Zero

**Purpose –** Ensure that the maxpect configuration works.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.14

### Test Procedure – Configure Maxpect

**Purpose –** Ensure that the maxpect configuration works.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.15

### Test Procedure – Configure Padx Single Value

**Purpose –** Ensure that the padx configuration works for a single value.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.16

### Test Procedure – Configure Padx Multiple Values

**Purpose –** Ensure that the padx configuration works for multiple values.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.17

### Test Procedure – Configure Pady Single Value

**Purpose –** Ensure that the pady configuration works for a single value.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.18

### Test Procedure – Configure Pady Multiple Values

**Purpose –** Ensure that the pady configuration works for multiple values.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.19

### Test Procedure – Configure Paper Height

**Purpose –** Ensure that the paperheight configuration works for valid values.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.20

### Test Procedure – Configure Paper Width

**Purpose –** Ensure that the paperwidth configuration works for valid values.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.21

### Test Procedure – Configure Width Zero

**Purpose –** Ensure that the width configuration works for zero.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.22

### Test Procedure – Configure Width

**Purpose –** Ensure that the width configuration works for valid values.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.2.23

### Test Case 3

**Test Case ID –** RBC.graph.postscript.3

**Test Item –** The *postscript output* command of the *graph* BLT component.

**Input Specification –** An existing graph, optional *filename*, and optional *option*-*value* pairs

**Output Specification –** Postscript representing the graph will be generated according to the configuration as dictated by the *option-value* pairs. If *filename* is present, the Postscript will be saved to the file, otherwise it will be returned to the console

**Special Procedural Requirements –** None

**Inter-case Dependencies –** None

### Test Procedure – Output to Console

**Purpose –** Ensure that output will print postscript to the console.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.3.1

### Test Procedure – Output to File

**Purpose –** Ensure that output will print postscript to a file.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.3.2

### Test Procedure – Output to Console with Options

**Purpose –** Ensure that output will print postscript to the console with option-value pairs.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.3.3

### Test Procedure – Output to File with Options

**Purpose –** Ensure that output will print postscript to a file with option-value pairs.

**Special Requirements –** None

**TclTest –** RBC.graph.postscript.3.4

### Test Case 4

**Test Case ID –** RBC.graph.postscript.4

**Test Item –** The *postscript output* command of the *graph* BLT component.

**Input Specification –** An existing graph, optional *filename*, and optional *option*-*value* pairs

**Output Specification –** Postscript representing the graph will be generated according to the configuration as dictated by the *option-value* pairs. If *filename* is present, the Postscript will be saved to the file, otherwise it will be returned to the console

**Special Procedural Requirements –** None

**Inter-case Dependencies –** None

### Test Procedure – Output to Console

**Purpose –** Ensure that correct postscript output is returned.

**Special Requirements –** Current working directory for the wish interpreter must be known.

**Procedural Steps**

* Setup – Run the “RBC.graph.postscript.4.tcl” file and then call the “graph.postscript::RBC.graph.postscript.4.1.Setup” Tcl command
* Pre-Condition – An empty graph exists with only the axes showing. There is a file “RBC.graph.postcript.4.1.ps” in the current working directory with a size of 0Kb.
* Body

1. Call the “graph.postscript::RBC.graph.postscript.4.1.Body” Tcl command

* Post-Condition – The file “RBC.graph.postscript.4.1.ps” has content in it that matches the content in the “postScriptOutputTest.ps” file
* Cleanup – Call the “graph.postscript::RBC.graph.postscript.4.1.Cleanup” command

### Test Procedure – Output to File

**Purpose –** Ensure that correct postscript output is written to a file.

**Special Requirements –** Current working directory for the wish interpreter must be known.

**Procedural Steps**

* Setup – Run the “RBC.graph.postscript.4.tcl” file and then call the “graph.postscript::RBC.graph.postscript.4.2.Setup” Tcl command
* Pre-Condition – An empty graph exists with only the axes showing.
* Body

1. Call the “graph.postscript::RBC.graph.postscript.4.2.Body” Tcl command

* Post-Condition – The file “RBC.graph.postscript.4.2.ps” has content in it that matches the content in the “postScriptOutputTest.ps” file
* Cleanup – Call the “graph.postscript::RBC.graph.postscript.4.2.Cleanup” command

# Graph Snap

### Test Case 1

**Test Case ID –** RBC.graph.snap.1

**Test Item –** The *snap* function of the *graph* BLT component.

**Input Specification –** A image to put BLT graph image in

**Output Specification –** The image contains the image of the graph

**Special Procedural Requirements –** None

**Inter-case Dependencies –** None

### Test Procedure – Full Sized

**Purpose –** Ensure snap takes a picture and stores it as a Tk image.

**Special Requirements –** None

**Script –** RBC.graph.snap.1.tcl

**Procedural Steps**

* Setup – Call the “graph.snap::RBC.graph.snap.1.1.Setup” Tcl command
* Pre-Condition – A graph is displayed with line data on it
* Body

1. Call the “graph.snap::RBC.graph.snap.1.1.Body” Tcl command

* Post-Condition – A button with a red background and image of the pre-condition graph on it.
* Cleanup – Call the “graph.snap::RBC.graph.snap.1.1.Cleanup” command

### Test Procedure – Vertically Smaller (Doesn’t Work)

**Purpose –** Ensure snap takes a set height picture and stores it as a Tk image.

**Special Requirements –** None

**Script –** RBC.graph.snap.1.tcl

**Procedural Steps**

* Setup – Call the “graph.snap::RBC.graph.snap.1.2.Setup” Tcl command
* Pre-Condition – A graph is displayed with line data on it
* Body

1. Call the “graph.snap::RBC.graph.snap.1.2.Body” Tcl command

* Post-Condition – A button with a red background and image of the vertically smaller pre-condition graph on it.
* Cleanup – Call the “graph.snap::RBC.graph.snap.1.2.Cleanup” command

### Test Procedure – Horizontally Smaller (Doesn’t Work)

**Purpose –** Ensure snap takes a set width picture and stores it as a Tk image

**Special Requirements –** None

**Script –** RBC.graph.snap.1.tcl

**Procedural Steps**

* Setup – Call the “graph.snap::RBC.graph.snap.1.3.Setup” Tcl command
* Pre-Condition – A graph is displayed with line data on it
* Body

1. Call the “graph.snap::RBC.graph.snap.1.3.Body” Tcl command

* Post-Condition – A button with a red background and image of the horizontally smaller pre-condition graph on it.
* Cleanup – Call the “graph.snap::RBC.graph.snap.1.3.Cleanup” command

# Graph Transform

### Test Case 1

**Test Case ID –** RBC.graph.transform.1

**Test Item –** The *invtransform* function of the *graph* BLT component.

**Input Specification –** A graph coordinate (X and Y)

**Output Specification –** The Window coordinates translated from the inputs

**Special Procedural Requirements –** None

**Inter-case Dependencies –** None

### Test Procedure – Transform

**Purpose –** Ensure the transform command works correctly.

**Special Requirements –** None

**Script –** RBC.graph.transform.1.tcl

**Procedural Steps**

* Setup – Call the “graph.transform::RBC.graph.transform.1.1.Setup” Tcl command
* Pre-Condition – A graph is displayed
* Body

1. Call the “graph.transform::RBC.graph.transform.1.1.Body” Tcl command

* Post-Condition – Command returns 5060 -1288
* Cleanup – Call the “graph.transform::RBC.graph.transform.1.1.Cleanup” command

# Graph Xaxis

### Test Cases

**Test Case ID –** RBC.graph.axis.\*

**Test Item –** The *xaxis* function of the *graph* BLT component.

**Input Specification –** See axis input specification

**Output Specification –** See axis output specification

**Special Procedural Requirements –** Use pathname xaxis operation instead of pathname axis operation xaxis

**Inter-case Dependencies –** See axis inter-case dependencies

# Graph X2axis

### Test Cases

**Test Case ID –** RBC.graph.axis.\*

**Test Item –** The *x2axis* function of the *graph* BLT component.

**Input Specification –** See axis input specification

**Output Specification –** See axis output specification

**Special Procedural Requirements –** Use pathname x2axis operation instead of pathname axis operation x2axis

**Inter-case Dependencies –** See axis inter-case dependencies

# Graph Yaxis

### Test Cases

**Test Case ID –** RBC.graph.axis.\*

**Test Item –** The *yaxis* function of the *graph* BLT component.

**Input Specification –** See axis input specification

**Output Specification –** See axis output specification

**Special Procedural Requirements –** Use pathname yaxis operation instead of pathname axis operation yaxis

**Inter-case Dependencies –** See axis inter-case dependencies

# Graph Y2axis

### Test Cases

**Test Case ID –** RBC.graph.axis.\*

**Test Item –** The *y2axis* function of the *graph* BLT component.

**Input Specification –** See axis input specification

**Output Specification –** See axis output specification

**Special Procedural Requirements –** Use pathname y2axis operation instead of pathname axis operation yaxis

**Inter-case Dependencies –** See axis inter-case dependencies