logo

QESimpleShape Widget

Andrew Starritt

15th November 2023

Copyright (c) 2023 Australian Synchrotron

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.  
A copy of the license is included in the section entitled "GNU Free Documentation License" within the QE\_QEGuiAndUserInterfaceDesign document.

Contents

[Introduction 3](#_Toc150955909)

[QSimpleShape 3](#_Toc150955910)

[Description 3](#_Toc150955911)

[Examples 4](#_Toc150955912)

[Properties 4](#_Toc150955913)

[QESimpleShape 5](#_Toc150955914)

[Description 5](#_Toc150955915)

[Properties 6](#_Toc150955916)

# Introduction

This document describes in detail the QSimpleShape and QESimpleShape widgets provided by the EPICS Qt, aka QE, Framework.

This document was created as a separate widget specification document. The main reason for this is ease of maintenance and avoiding editing large and unwieldly word documents.

The QE Framework is distributed under the GNU Lesser General Public License version 3, distributed with the framework in the file LICENSE. It may also be obtained from here: <http://www.gnu.org/licenses/lgpl-3.0-standalone.html>

# QSimpleShape

## Description

The QSimpleShape is essentially a non-EPICS aware widget that can display different shapes in various colours and with text. The colour may be one of 16 property specified colours and the text may be fixed, or one of 16 property specified state values. It also has two text selection modes to allow the QESimpleShape class to provide the text.

The decision to provide up to 16 colours was some-what arbitrary; and while a user can only readily identify a limited number of colours (as opposed to distinguishing between subtle shade differences presented side by side) 16 was chosen as a nod to QESimpleShape class so that a colour could be associated with each value of an mbbi/mbbo record.

The shape itself is determined by the widget’s shape property, and may be one of:   
noShape, circle, ellipse, rectangle (default), roundRectangle, roundSquare, square, triangleUp, triangleDown, triangleLeft, triangleRight, triangleTopRight, triangleBottomRight, triangleBottomLeft, triangleTopLeft, diamond, equalDiamond, arrowUp, arrowDown, arrowLeft, arrowRight, crossHorizontal, crossVertical, hexagon, octogon, snakeHorizontal, snakeVertical, pentagon, star, plus, tick, cross, roundpie, pie, heart.

The size of the shape is maximised to just fit within the geometry of the widget. For circle, square, roundSquare, equalDiamond and roundpie the size is determined by the lesser of the widget’s width and height.

The QESimpleShape also has an edge, the width and colour of which are specified by widget properties.

## Examples

A screenshot of a computer

Description automatically generated

Figure 1 QSimpleShape examples

The above figure show example of the shapes in various colours, and a few with text.

## Properties

The QSimpleShape has the following properties.

|  |  |  |  |
| --- | --- | --- | --- |
| **name** | **type** | **default** | **description** |
| shape | enum | rectangle | Specifies the shape – see list above. |
| edgeWidth | int | 1 | With of the boarder – limited 0 to 20. |
| edgeStyle | eum | SolidLine | Pen style used to draw the boarder. |
| semiCycles | int | 8 | Applies only to the snakeHorizontal and snakeVertical. Limited to 1 to 30. |
| percentSize | int | 10 | Applies only to  Limited to 1 to 50 |
| centreAngle | int | 0 |  |
| spanAngle | int | 90 |  |
| value | int | 0 | Controls the statue, i.e. the colour and the text used to draw the QESimpleShape. The value is limited to 0 to the modulus value. |
| modulus | int | 16 | Defines the modulus value. Limited 2 to 16 |
| format | enum | FixedText | Allowed values are:  FixedText - text provided with the *fixedText* property. StateSet - text selected from the *stateSet* property. PVText - only applies to QESimpleShate. LocalEnumeration - only applies to QESimpleShate. |
| fixedText | string |  | Defines the fixed text |
| stateSet | string list |  | Defines upto 16 state values. Selected text depends on the *value* and *modulus* properties. |
| alignment | enums | AlignHCenter, AlignVCenter | Controls the text alignment with the widget. |
| indent | int | 6 | Control the text indent (just like QLabel) |
| flashRate | enum | Medium | Allowed values are Very Slow, Slow, Medium, Fast and VeryFast. Controls rate at which the widget alternates between the *flashOffColour* and the normal colour. These currently correspond to flashing rates of 0.25Hz, 0.5Hz, 1Hz, 2Hz and 4Hz respectively. |
| isActive | bool | true | When *isActive* is unchecked/set false, all colours are greyed out |
| edgeColour | colour | black | This defines the colour of the shape’s boundary |
| flashOffColour | colour | grey, alpha=0 | Specified the alternative colour when flashing |
| colour*N* | colour | grey | *N* is in the range 0 to 15.  These 16 properties assign the colour used.  The selected colour depends on the *value* and *modulus* properties. |
| flash*N* | bool | false | *N* is in the range 0 to 15.  These 16 properties indicate if flashing applied to the selected state. |

# QESimpleShape

## Description

The QESimpleShape inherits directly from QSimpleShape. The QESimpleShape widget is an EPICS aware widget which uses either the alarm state or the value of a single PV to determine the colour of the shape. It displays alarm state by default.

When the *displayAlarmStateOption* property is set to ‘Always’ (the default) or is set to ‘WhenInAlarm’ and the PV is in an alarm state, the colour of the widget is determined by the alarm state of the PV. Standard framework alarm colours are used, i.e. green for no alarm, yellow for minor alarm, red for major alarm and white for invalid alarm.

When the *displayAlarmStateOption* property is set ‘Never’, the value of the PV is used to select a colour from a set of 16 colour properties, i.e. color0, colour1, and so on to colour15. The value of the PV must be capable of being interpreted as an integer. Modulo arithmetic is used to ensure the PV value yields a number in the range. The modulus property (range 2 to 16, default 16) defines the modulo arithmetic behaviour. The widget has an arrayIndex property that can be used to select a single element from an array of data to provide the state value. The default array index value is 0.

All states that are flashing use the same flash rate and the same flash off colour. Even when the *displayAlarmStateOption* property is ‘Always’ or is ‘WhenInAlarm’ and the variable is in an alarm state, i.e. the colour being derived from the PV alarm state, the is-flashing state is determined from the PV value. If flashing or not flashing by alarm state is required, one option would be to monitor a record’s SEVR field.

## Properties

|  |  |  |  |
| --- | --- | --- | --- |
| **name** | **type** | **default** | **description** |
| variable | string |  |  |
| variableSubstitution | string |  | Provided lowest-priority default macro substitutions |
| elementsRequired | int | 0 |  |
| arrayIndex | int | 0 |  |
| edgeVariable | string |  |  |
| edgeElementsRequired | int | 0 |  |
| edgeArrayindex | int | 0 |  |
| edgeAlarmStateOption | enum | Always | This is one Always, WhenInAlarm, WhenInvalid, Never. Standard property – see general widget documentation and not below. |
| addUnits | bool | true | Controls if engineering units are appended to the PV text (provided that *format* is PVText). |
| localEnumeration | string |  | Provides the local enumeration values (provided that *format* is LocalEnumeration). |
| variableAsToolTip | bool | true | Standard framework property |
| allowDrop | bool | false | Standard framework property |
| visible | bool | true | Standard framework property |
| messageSourceId | int | 0 | Standard framework property |
| defaultStyle | string |  | Standard framework property. However not applicable. |
| userLevelUserStyle | string |  | Standard framework property. However not applicable. |
| userLevelScientistStyle | string |  | Standard framework property. However not applicable. |
| userlevelEngineerStyle | string |  | Standard framework property. However not applicable. |
| userLevelVisibility | enum | User | Standard framework property. |
| userLevelEnabled | enum | User | Standard framework property. |
| displayAlarmStateOption | enum | Always | This is one Always, WhenInAlarm, WhenInvalid, Never. Standard property – see general widget documentation and not below. |
| oosAware | bool | true | Standard framework property. |
| useStyleAlarmColours | bool | false | When set true, the alarm colours used are the paler, less intense colours as used by QELabel. |

#### Notes

edgeAlarmStateOption and displayAlarmStateOption control if/when the alarm state colour is used as opposed to the state-based colour when displaying the widget. At one extreme, Always means that only alarm state colours will ever be used; and at the other, Never means that alarm state colours will never be user. If running in mixed mode, e.g. WhenInvalid, it is good to ensure state-based colours are not confused with alarm-based colours.