



QELineEdit Widget

Zai Wang

10th July 2020

Copyright (c) 2020 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
Description	3
Properties.....	4
subscribe: bool.....	5
writeOnLoseFocus: bool	5
writeOnEnter: bool	5
writeOnFinish: bool.....	5
writeOnFinish: bool.....	5
allowFocusUpdate: bool	5
dropOption: DropOption	5

QLEdit Widget Specification



Introduction

This document describes in detail the QLEdit widget which is an EPICS aware widget 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 unwieldy 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>

Description

The QLEdit widget provides the ability to textually modify the value of a single PV. This widget is (indirectly) derived from QLineEdit. The example in Figure shows a QLEdit widget connected to an ao record. While this widget is primarily intended for writing to string PVs, it can also be used with numerical PVs as in this example. However, in this case, a **Error! Reference source not found.** or a **Error! Reference source not found.** widget may be more appropriate.

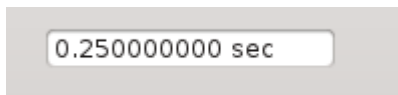
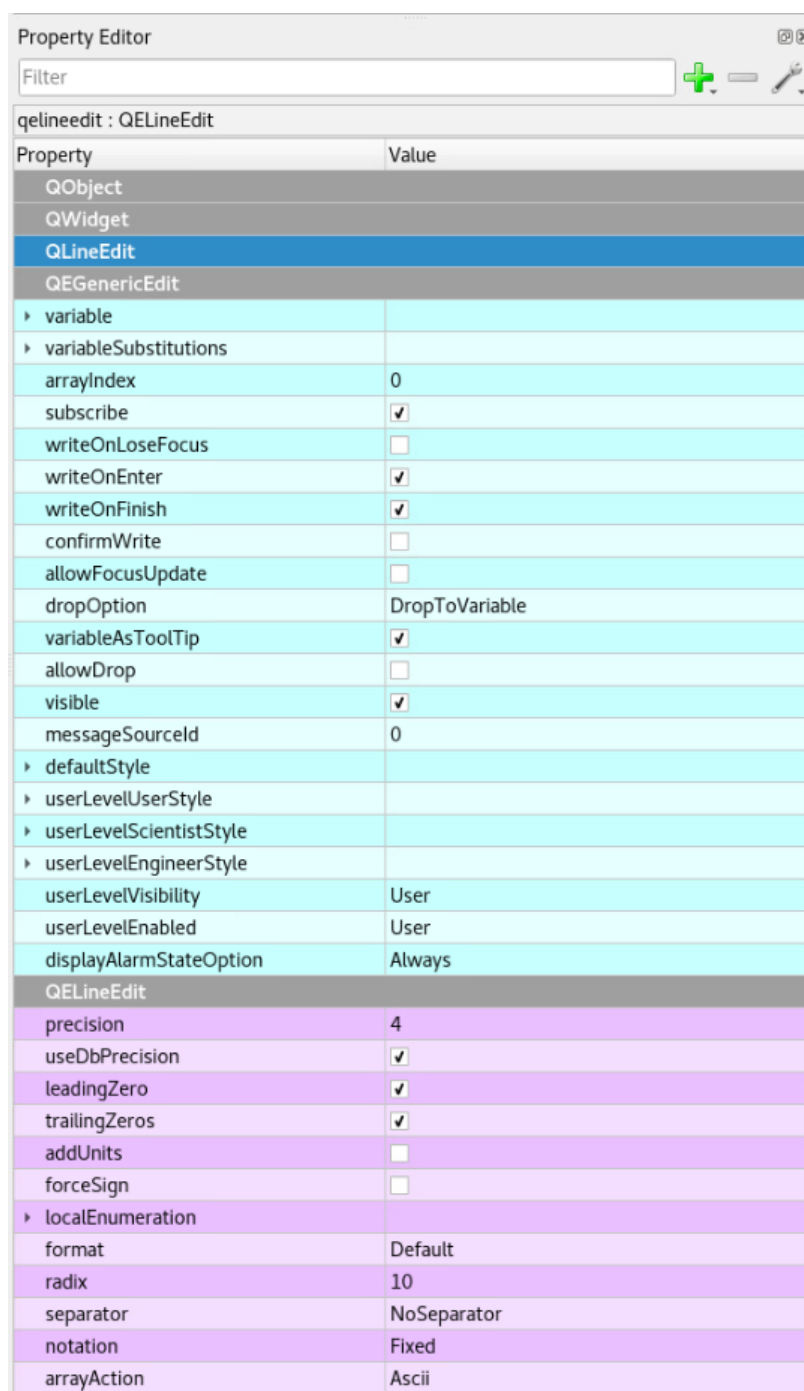


Figure 1 QLEdit example

QLineEdit Widget Specification

Properties

The behaviour of the widget is defined by the widget specific properties as shown in Figure .



The screenshot shows the Qt Property Editor window for the QLineEdit widget. The window has a title bar 'Property Editor' and a search filter field. Below the filter, a tree view shows the inheritance hierarchy: QObject, QWidget, QLineEdit (selected), and QEGenericEdit. The main area displays a list of properties and their values. Properties are grouped into sections: QEGenericEdit (variable, variableSubstitutions, arrayIndex, subscribe, writeOnLoseFocus, writeOnEnter, writeOnFinish, confirmWrite, allowFocusUpdate, dropOption, variableAsToolTip, allowDrop, visible, messageSourceId, defaultStyle, userLevelUserStyle, userLevelScientistStyle, userLevelEngineerStyle, userLevelVisibility, userLevelEnabled, displayAlarmStateOption), QLineEdit (precision, useDbPrecision, leadingZero, trailingZeros, addUnits, forceSign, localEnumeration, format, radix, separator, notation, arrayAction).

Property	Value
QObject	
QWidget	
QLineEdit	
QEGenericEdit	
▶ variable	
▶ variableSubstitutions	
arrayIndex	0
subscribe	<input checked="" type="checkbox"/>
writeOnLoseFocus	<input type="checkbox"/>
writeOnEnter	<input checked="" type="checkbox"/>
writeOnFinish	<input checked="" type="checkbox"/>
confirmWrite	<input type="checkbox"/>
allowFocusUpdate	<input type="checkbox"/>
dropOption	DropToVariable
variableAsToolTip	<input checked="" type="checkbox"/>
allowDrop	<input type="checkbox"/>
visible	<input checked="" type="checkbox"/>
messageSourceId	0
▶ defaultStyle	
▶ userLevelUserStyle	
▶ userLevelScientistStyle	
▶ userLevelEngineerStyle	
userLevelVisibility	User
userLevelEnabled	User
displayAlarmStateOption	Always
QLineEdit	
precision	4
useDbPrecision	<input checked="" type="checkbox"/>
leadingZero	<input checked="" type="checkbox"/>
trailingZeros	<input checked="" type="checkbox"/>
addUnits	<input type="checkbox"/>
forceSign	<input type="checkbox"/>
▶ localEnumeration	
format	Default
radix	10
separator	NoSeparator
notation	Fixed
arrayAction	Ascii

Figure 2 QLineEdit properties

QLineEdit Widget Specification

As well as the usual PV variable name, substitutions, display format, user level etc. properties, the widget has additional properties to control its mode of operation:

subscribe: bool

This determines if the widget subscribes for data updates and displays current data.

default value: true

writeOnLoseFocus: bool

This defines when true this widget automatically writes any changes when it loses focus.

default value: false

writeOnEnter: bool

This defines when true writes when the user presses 'enter'. Note, the current value will be written even if the user has not changed it.

default value: true

writeOnFinish: bool

This defines when true writes any changes when the user finished editing (the QLineEdit 'editingFinished' signal is emitted). No writing occurs if no changes were made.

default value: true

writeOnFinish: bool

This defines when true this widget will ask for confirmation (using a dialog box) prior to writing data focus.

default value: false

allowFocusUpdate: bool

This defines when true this widget update even if the widget currently has focus.

default value: false

dropOption: DropOption

Date type definition: enum DropOption { DropToVariable, DropToText, DropToTextAndWrite }

This defines the behaviour of drag and drop text to the QLineEdit widget.

DropToVariable: The dropped text will be used as a new PV name.

DropToText: The dropped text will be written on the widget.

QELineEdit Widget Specification



DropToTextAndWrite: The dropped text will be written on the widget and to the defined PV as a value.

default value: DropToVariable