



QEPvLoadSave Specification

Andrew Starritt

23rd November 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
QEPvLoadSave.....	3
Description.....	3
Tool Bar	4
Context Menu	6
Drop	7
XML File Format	7
Merged Node Names	8
Properties.....	9
configurationFileLeft : QString.....	9
configurationFileRight : QString.....	9
defaultSubstitutions : QString.....	9
defaultDir : QString	9
confirmAction : bool	9
QEPvLoadSaveButton.....	10
Description	10
Properties.....	10
configurationFile : QString	10
defaultSubstitutions : QString.....	10
action : enumeration	10
confirmAction : bool	10
confirmText : QString	10
showProgressBar : bool	10

Introduction

This document describes in detail the related QEPvLoadSave and QEPvLoadSaveButton widgets (strictly speaking, we should call these widget classes, but the term “widget” is often used through-out this document) which are two EPICS aware 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 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>

QEPvLoadSave

Description

The QEPvLoadSave widget is designed and provided primarily to support the in built-in PV Load/Save form included in the QEGui application. However form designers may include one or more instances of this widget on their own forms if so desired.

The QEPvLoadSave widget allows a user to define a hierarchical set of variables and apply the following actions to the whole hierarchy or a selected subset:

- a) Write the values to the ‘system’ – the system being whatever IOCs and other Channel Access servers are currently accessible to the QEGui application;
- b) Read the values from the system;
- c) Read the values from the archive for a user nominated time;
- d) Write the values to a file for not volatile storage. The file is an xml file - the format is described below;
- e) Read the values from a file; and
- f) Edit a nominated value.

The QEPvLoadSave widget can supports two simultaneous and independent hierarchies, and the user is able to merge the whole hierarchy or a selected subset into the other hierarchy. The user may also request the display of the difference between the hierarchies. This is presented graphically to the user to enable him/her to quickly identify differences in the values associated between the PVs common to both sets.

Figure 1 below shows the QEPvLoadSave widget as used within the QEGui built-in form.

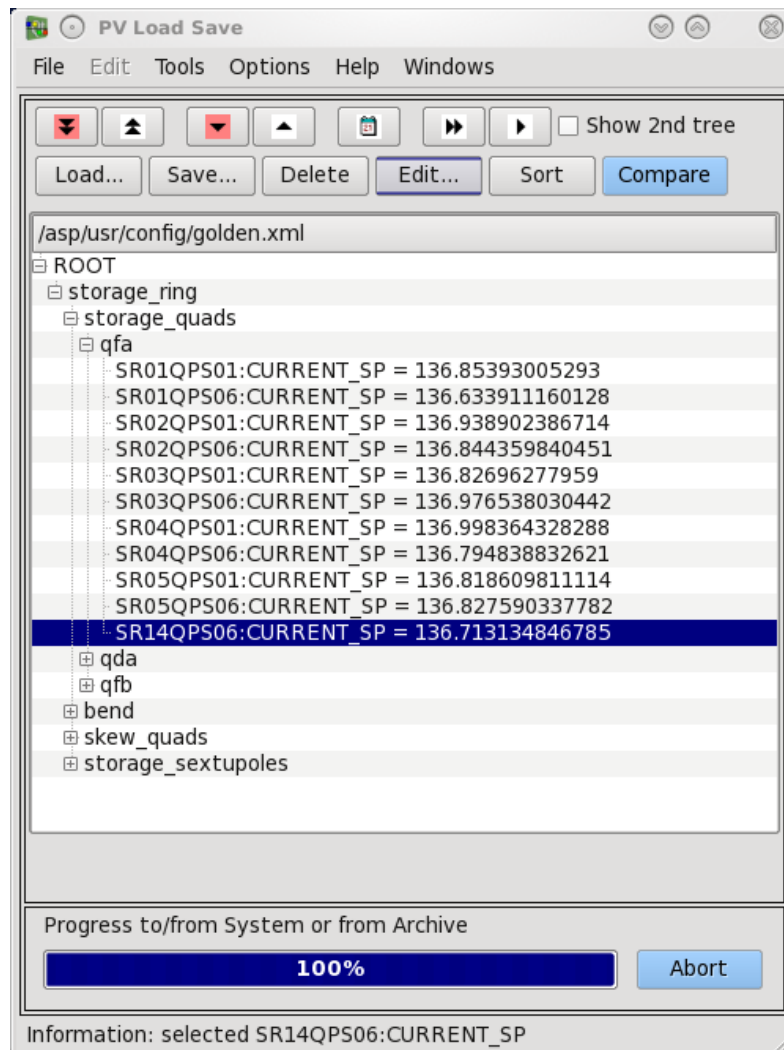











Figure 1 QEPvLoadSave – basic example.

Tool Bar

Each hierarchy is provided with a tool bar. The functions provided by each of these buttons are:

- a)  - this button writes all values in the hierarchy to their associated PVs;
- b)  - this button reads all values in the hierarchy from their associated PVs;
- c)  - this button writes the selected sub-hierarchy values to their associated PVs;
- d)  - this button reads the selected sub-hierarchy values from their associated PVs;

- e)  - this button displays a date/time selection dialog. Once the user has selected a data and time, the archiver is accessed and the associated values extracted;
- f)  or  - this button copies all values from the hierarchy and merges these into the other hierarchy;
- g)  or  - this button copies all values from the selected sub-hierarchy and merges these into the other hierarchy;
- h) Show second tree check box – this control whether the second tree (hierarchy) is displayed;
- i) Load... - this button allows the user to navigate the file system to load a PV-Value xml file.
- j) Save... - this button allows the user to navigate the file system and select the file to save the current configuration file;
- k) Delete - this button delete the selected sub-hierarchy;
- l) Edit... - this button allows the user to edit the value of the selected PV, or values for a an array PV;
- m) Sort - this button sort by PV name the selected sub-hierarchy (*this functionality is TDB*); and
- n) Compare - this button generates a graphical comparison of the two sets of PVs and their associated values. Only numerical values common to both hierarchies contribute to this graphical display .

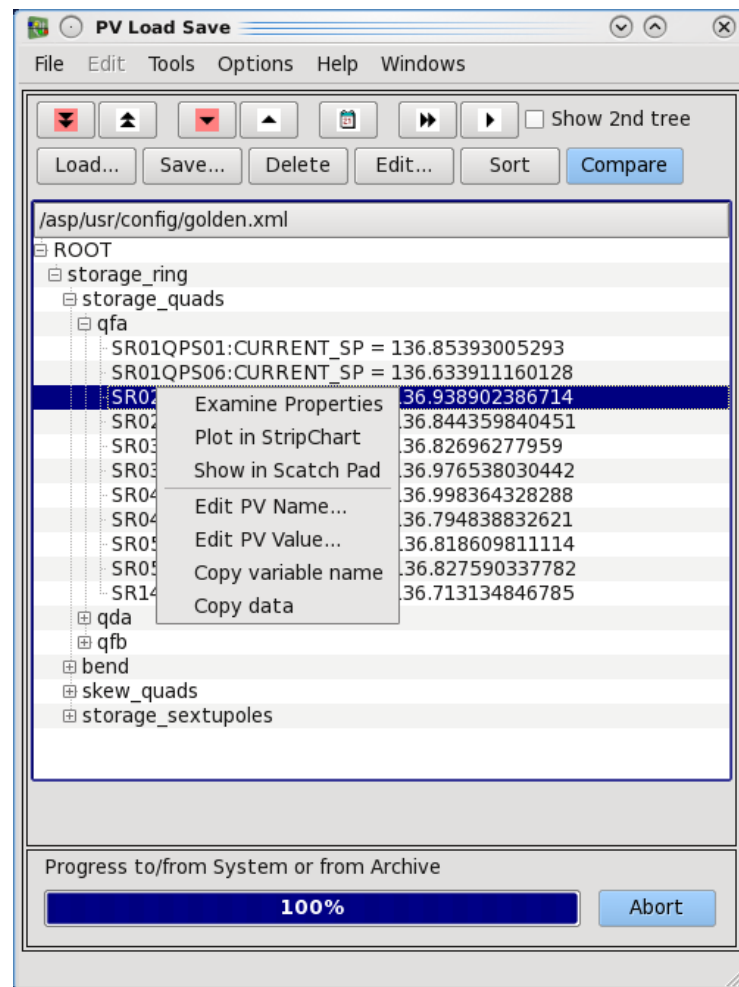


Figure 2 QEPvLoadSave – context menu example.

Context Menu

The QEPvLoadSave widget tree hierarchies provided context menus to allow the following. The content of the context menu depends on the type of item if any selected when the context menu is launched. Figure 2 above shows the context menu presented to the user when a PV node in the hierarchy is selected. Most of these items mirror those available for any EPICS aware framework widget. QEPvLoadSave specific context menu items are:

- Create Root (not shown in example). This is only available for an empty hierarchy and creates the root node. A root node is currently required;
- Add Group... (not shown in example). This allows a group to be added to the hierarchy. It is only available if the root node or another group node is selected;
- Rename Group... (not shown in example). This allows the group name to be modified. It is only available if a group node is selected;

- d) Add PV... (not shown in example). This allows a PV to be added to the hierarchy. It is only available if the root node or another group node is selected;
- e) Edit PV Name... This allows the PV name to be modified. It is only available if a PV node is selected;
- f) Edit PV Value.... This allows the user to edit the value(s) of the selected PV - is essentially the same as using the edit button as described above.

Drop

The PV name from another framework widget may be dragged and dropped onto any group node on the hierarchy. If the dropped onto node is a group node, the new PV name is added to the end of the group. If the dropped onto node is a PV node, the new PV name becomes a sibling of that node, i.e. this is as if it had been dropped on the PV node's parent group node.

Note: currently one cannot drag from or between the hierarchy trees.

XML File Format

The format of the xml file used to store the hierarchy in a file is illustrated by example in Figure 3 below.

QEPvLoadSave: This tag defines the file type and the format version number.

Group: The group tag defines the hierarchy, it can be nested to any arbitrary depth. The "Name" attribute defines the group name.

PV: This tag specifies a scalar PV. There are one obligatory attributes, namely "Name" which defines the PV name. Other attributes are "Value" which defines the value, "ReadPV" which specifies the PV to read from if different from "Name" which is the which PV written to. The attribute "ArchPV" defines the value read from the archiver if different from "Name".

Array: This tag specifies an array PV. The "Name", "ReadPV" and "ArchPV" attributes are similar to those of the *PV* tag. This "Number" attribute specified the number of element values.

Element: This tag specifies an array element. There are two attributes, namely "Value" which defines the array element value and "Index" is specifies the elements position, counting from zero, in the array.

Note: The "Number" and "Index" attributes are probably overkill and may be dropped in version 2 of the xml file format.

```

<QEPvLoadSave Version="1">
  <Group Name="Foo">
    <PV Value="19.7" Name="SR11SCR01:UPPER_MOTOR.VAL"
      ReadPV="SR11SCR01:UPPER_MOTOR.RBV"
      ArchPV="SR11SCR01:UPPER_MOTOR"/>
  </Group>
  <Group Name="Bar">
    <Array Number="14" Name="SR00BLM00:ACTIVE_MAP">
      <Element Value="113" Index="0"/>
      <Element Value="114" Index="1"/>
      <Element Value="201" Index="2"/>
      <Element Value="202" Index="3"/>
      <Element Value="203" Index="4"/>
      <Element Value="304" Index="5"/>
      <Element Value="305" Index="6"/>
      <Element Value="406" Index="7"/>
      <Element Value="407" Index="8"/>
      <Element Value="508" Index="9"/>
      <Element Value="509" Index="10"/>
      <Element Value="712" Index="11"/>
      <Element Value="810" Index="12"/>
      <Element Value="811" Index="13"/>
    </Array>
  </Group>
</QEPvLoadSave>

```

Figure 3 QEPvLoadSave – xml file example.

Merged Node Names

If/when a different PV name is used for writing, reading or extracting archive data node name on display is based on all three PVs names. This consists of:

```
<common_part>{w:<write_part>; r:<read_part>; a:<arch_part>;}
```

For the example above, this would be:

```
SR11SCR01:UPPER_MOTOR{w:.VAL;r:.RBV}
```

When editing the PV name, the 'w', 'r' and 'a' parts may be specified in any order, and if two "parts" are the same, then can be entered and will be displayed together, for example:

```
<common_part>{wa:<combined_part>; ... }
```


Properties

The QEPvLoadSave widget inherits directly from QEFrame which provides many of the properties common to many EPICS Qt framework widgets.

configurationFileLeft : QString

default: empty string

This defines the name of the xml configuration file to be loaded into the left hand side hierarchy. Like process variable names, substitutions are allowed here and the usual designer 1 second delay after editing before action delay applies.

configurationFileRight : QString

default: empty string

This defines the name of the xml configuration file to be loaded into the right hand side hierarchy.

defaultSubstitutions : QString

default: empty string

This defines the default substitutions valued used by the widget. This are the lowest priority substitutions and will typically be overridden are run time.

defaultDir : QString

default: empty string

This defines the default file system directory used to load file from and save files to.

confirmAction : bool

default: true

This controls whether a confirmation dialog is presented to the user when he/she attempts to write values to any PVs.

QEPvLoadSaveButton

Description

The QEPvLoadSaveButton widget has the same basic functionality as the QEPvLoadSave widget but is an express version of it. Users can define an XML file, which has to have the same structure as the one used with QEPvLoadSave (see example in Figure 3 above), and the action they want the button to perform (Load PV values from file to the system, or read PVs from system to the file) and then load or save PVs using a single click.

Properties

The QEPvLoadSaveButton has the following custom properties

configurationFile : QString

default: empty string

This defines the absolute pathname to the load/save XML configuration file. This file should follow the format described in the by example in Figure 3 above. configurationFile property supports macros.

defaultSubstitutions : QString

default: empty string

This defines the default substitutions for the macro defined in configurationFile.

action : enumeration

allowed values: LoadToPVs, SaveToFile

default: LoadToPVs

This selects one of the two behaviours this button can have.

confirmAction : bool

default: false

This controls whether a confirmation dialog is presented to the user when he/she attempts to write values to any PVs.

confirmText : QString

default: "Do you want to perform this action?"

This defines the question asked in the confirmation dialog.

showProgressBar : bool

default: true

This controls the display a progress modal dialog while the action is being executed.