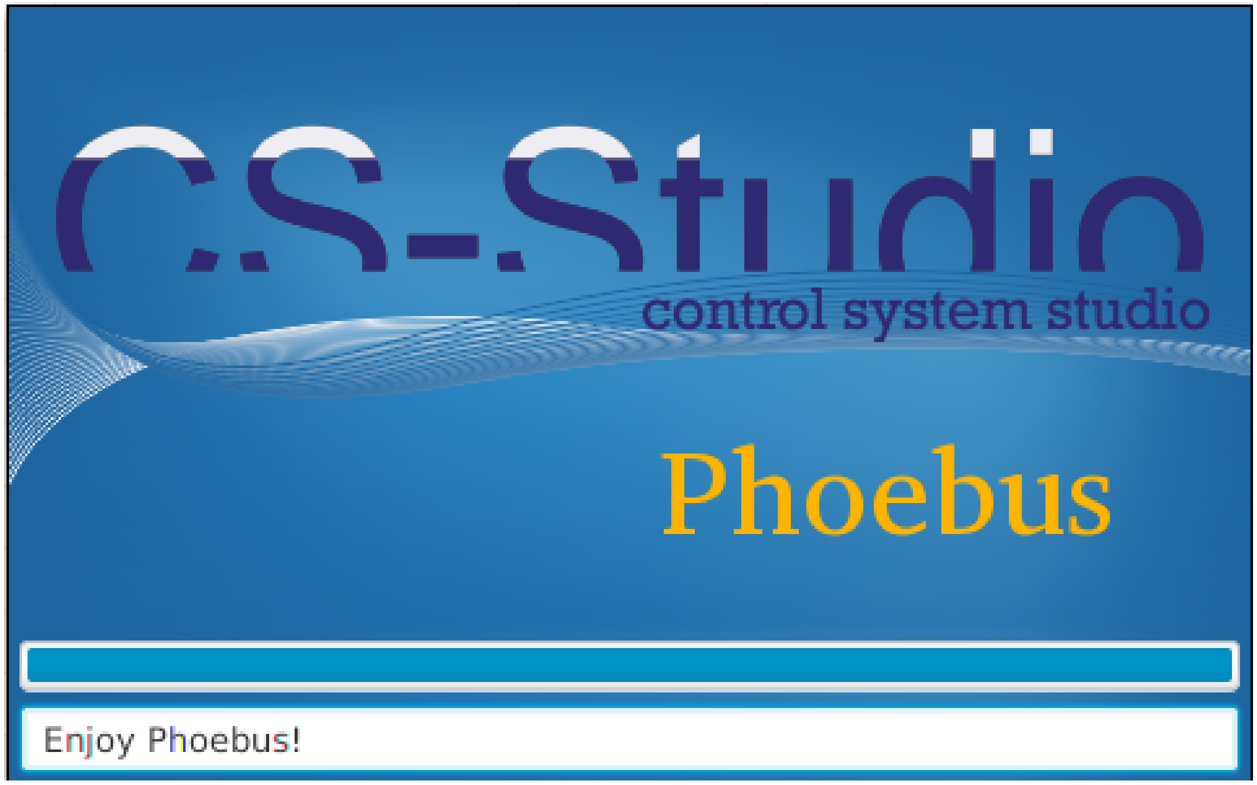
# Phoebus Documentation

*Release 1.0*

Phoebus Developers

May 24, 2023

# Contents

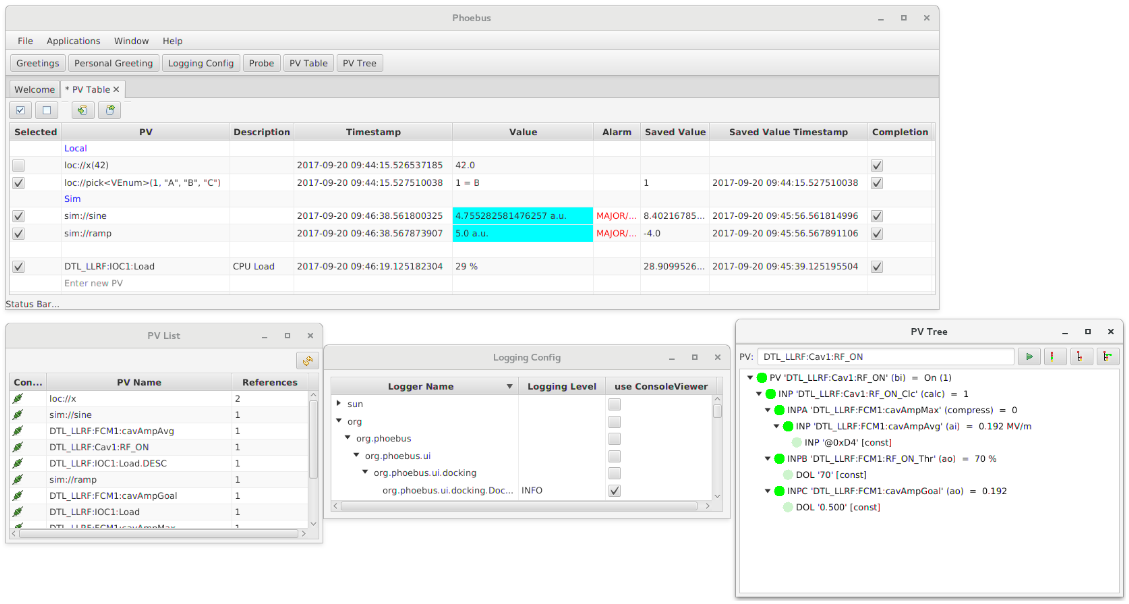


User Documentation:

CHAPTER 1

# Introduction

Phoebus is an update of the Control System Studio toolset that removes dependencies on Eclipse RCP and SWT.



| Selected | PV | Description | Timestamp | Value | Alarm | Saved Value | Saved Value Timestamp | Completion |
|---|---|---|---|---|---|---|---|---|
| | Local | | | | | | | |
| ☐ | loc://x(42) | | 2017-09-20 09:44:15.526537185 | 42.0 | | | | ✓ |
| ✓ | loc://pick<VEnum>(1, "A", "B", "C") | | 2017-09-20 09:44:15.527510038 | 1 = B | | 1 | 2017-09-20 09:44:15.527510038 | ✓ |
| | Sim | | | | | | | |
| ✓ | sim://sine | | 2017-09-20 09:46:38.561800325 | 4.755282581476257 a.u. | MAJOR/... | 8.40216785... | 2017-09-20 09:45:56.561814996 | ✓ |
| ✓ | sim://ramp | | 2017-09-20 09:46:38.567873907 | 5.0 a.u. | MAJOR/... | -4.0 | 2017-09-20 09:45:56.567891106 | ✓ |
| | | | | | | | | |
| ✓ | DTL_LLRF:IOC1:Load | CPU Load | 2017-09-20 09:46:19.125182304 | 29 % | | 28.9099526... | 2017-09-20 09:45:39.125195504 | ✓ |
| | Enter new PV | | | | | | | |

| Con... | PV Name | References |
|---|---|---|
| | loc://x | 2 |
| | sim://sine | 1 |
| | DTL_LLRF:FCM1:cavAmpAvg | 1 |
| | DTL_LLRF:Cav1:RF_ON | 1 |
| | DTL_LLRF:IOC1:Load.DESC | 1 |
| | sim://ramp | 1 |
| | DTL_LLRF:FCM1:cavAmpGoal | 1 |
| | DTL_LLRF:IOC1:Load | 1 |

| Logger Name | Logging Level | use ConsoleViewer |
|---|---|---|
| sun | | ☐ |
| org | | ☐ |
| org.phoebus | | ☐ |
| org.phoebus.ui | | ☐ |
| org.phoebus.ui.docking | | ☐ |
| org.phoebus.ui.docking.Doc... | INFO | ✓ |

```
PV: DTL_LLRF:Cav1:RF_ON
PV 'DTL_LLRF:Cav1:RF_ON' (bi) = On (1)
  INP 'DTL_LLRF:Cav1:RF_ON_Clc' (calc) = 1
    INPA 'DTL_LLRF:FCM1:cavAmpMax' (compress) = 0
      INP 'DTL_LLRF:FCM1:cavAmpAvg' (ai) = 0.192 MV/m
        INP '@0xD4' [const]
    INPB 'DTL_LLRF:FCM1:RF_ON_Thr' (ao) = 70 %
      DOL '70' [const]
    INPC 'DTL_LLRF:FCM1:cavAmpGoal' (ao) = 0.192
      DOL '0.500' [const]
```

While Eclipse RCP kick-started the original CS-Studio implementation and served CS-Studio well for about a decade, depending on RCP also added limitations to the control system user interface development.

Goals of the Phoebus project:

- Retain functionality of key CS-Studio tools, specifically the Display Builder, Data Browser, PV Table, PV Tree, Alarm UI, Scan UI, .. supporting their original configuration files with 100% compatibility.
- Provide full control of window placement free from RCP restrictions, for example allowing us to save/restore panel layouts.
- Use Java FX as the graphics library to overcome limitations of SWT.

- Prefer core Java functionality over external libraries whenever possible: Java FX as already mentioned, SPI for locating extensions, java.util for logging and preferences. In the future, we may also use the module mechanism introduced in Java 9 for bundling.
- Reduce build system complexity, fetching external dependencies in one initial step, then supporting a fully standalone, reproducible build process, allowing multiple build methods instead of being restricted to one.

For more, see https://docs.google.com/document/d/11W52PRlsRjpIvP81HxUxxR9g180DHDByCohYQ9TQv7U

# CHAPTER 2

# Starting CS-Studio/Phoebus

For build instructions, refer to the README.md on https://github.com/ControlSystemStudio/phoebus

For pre-built binaries, see https://controlssoftware.sns.ornl.gov/css_phoebus/

From the command-line, invoke `phoebus.sh -help`, which will look similar to this, but check your copy of CS-Studio/Phoebus for the complete list:

```
 _______           _______  _______  ______            _______
(  ____ )|\     /|(  ___  )(  ____ \(  ___ \ |\     /|(  ____ \
| (    )|| )   ( || (   ) || (    \/| (   ) )| )   ( || (    \/
| (____)|| (___) || |   | || (__    | (__/ / | |   | || (_____
|  _____)|  ___  || |   | ||  __)   |  __ (  | |   | |(_____  )
| (      | (   ) || |   | || (      | (  \ \ | |   | |      ) |
| )      | )   ( || (___) || (____/\| )___) )| (___) |/\____) |
|/       |/     \|(_______)(_______/|/ \___/ (_______)\_______)

Command-line arguments:

   -help                                     -  This text
   -splash                                   -  Show splash screen
   -nosplash                                 -  Suppress the splash screen
   -settings settings.xml                    -  Import settings from file, either␣
↪exported XML or property file format
   -export_settings settings.xml             -  Export settings to file
   -logging logging.properties               -  Load log settings
   -list                                     -  List available application features
   -server port                              -  Create instance server on given TCP port
   -app probe                                -  Launch an application with input␣
↪arguments
   -resource  /tmp/example.plt               -  Open an application configuration file␣
↪with the default application
   -layout /path/to/Example.memento          -  Start with the specified saved layout␣
↪instead of the default 'memento'
   -clean                                    -  Start with a blank workspace. Overrides␣
↪-app, -resource and -layout.
```

(continues on next page)

(continued from previous page)

```
   -main org.package.Main                    -  Run alternate application Main
```

## 2.1 Command Line Parameters for Applications

To open an application feature like "probe" or the "pv_tree" from the command line, use the following example parameters.

Open empty instance of probe:

```
phoebus.sh -app probe
```

Open empty PV Table:

```
phoebus.sh -app pv_table
```

Open a file with the appropriate application feature (PV Table in this case):

```
phoebus.sh -resource "/path/to/example.pvs"
```

The '-resource' parameter can be a URI for a file or web link:

```
phoebus.sh -resource "http://my.site/path/to/example.pvs"
```

Some resource types are supported by multiple applications. For example, a display file "my_display.bob" can be handled by both the "display_runtime" and the "display_editor" application. A preference setting "org.phoebus.ui/default_apps" defines which application will be used by default, and a specific application can be requested like this:

```
phoebus.sh -resource "/path/to/my_display.bob?app=display_editor"
```

The schema 'pv://?PV1&PV2&PV3' is used to pass PV names, and the 'app=..' query parameter picks a specific app for opening the resource.

Since such resource URLs can contain characters like *&* that would also be interpreted by the Linux shell, best enclose all resources in quotes.

Open probe with a PV name:

```
phoebus.sh -resource "pv://?sim://sine&app=probe"
```

Open PV Table with some PVs:

```
phoebus.sh -resource "pv://?MyPV&AnotherPV&YetAnotherPV&app=pv_table"
```

Note that all these examples use the internal name of the application feature, for example "pv_table", and not the name that is displayed the user interface, like "PV Table". Use the `-list` option to see the names of all available application features.

Start with a specific layout:

```
phoebus.sh -layout /path/to/mylayout.memento
```

Restores the layout saved in a (non-default) memento file. User may create such a file from menu option *Window -> Save Layout As…*

Start with clean workspace:

```
phoebus.sh -clean
```

This will suppress restore of the layout saved in the default memento file in order to start the application with an empty workspace. If specified, `-resource`, `-app` and `-layout` will be ignored.

## 2.2 Server Mode

By default, each invocation of `phoebus.sh ...` will start a new instance, with its own main window etc.

In a control room environment it is often advantageous to run only one instance on a given computer. For this scenario, invoke `phoebus.sh` with the `-server` option, using a TCP port that you reserve for this use on that computer, for example:

```
phoebus.sh -server 4918
```

The first time you start phoebus this way, it will actually open the main window. Follow-up invocations, for example:

```
phoebus.sh -server 4918 -resource "/path/to/some/file.pvs"
```

will contact the already running instance and have it open the requested file.

# CHAPTER 3

# Window Environment

When you start Phoebus for the first time, it opens a main window. As you use the *Applications* menu to open for example Probe, the PV Tree etc., these all open up as new Tabs in the original window.

## 3.1 Tabs and Windows

The behavior of these tabs is very similar to the handling of tabs in a web browser. You can rearrange the tabs within a window by dragging them around. You can also drag a tab out of the window to detach it into its own window. Alternatively, you can invoke the *Detach* option from the context menu of the tab to detach it. Tabs can be dragged between the main window and such detached windows.

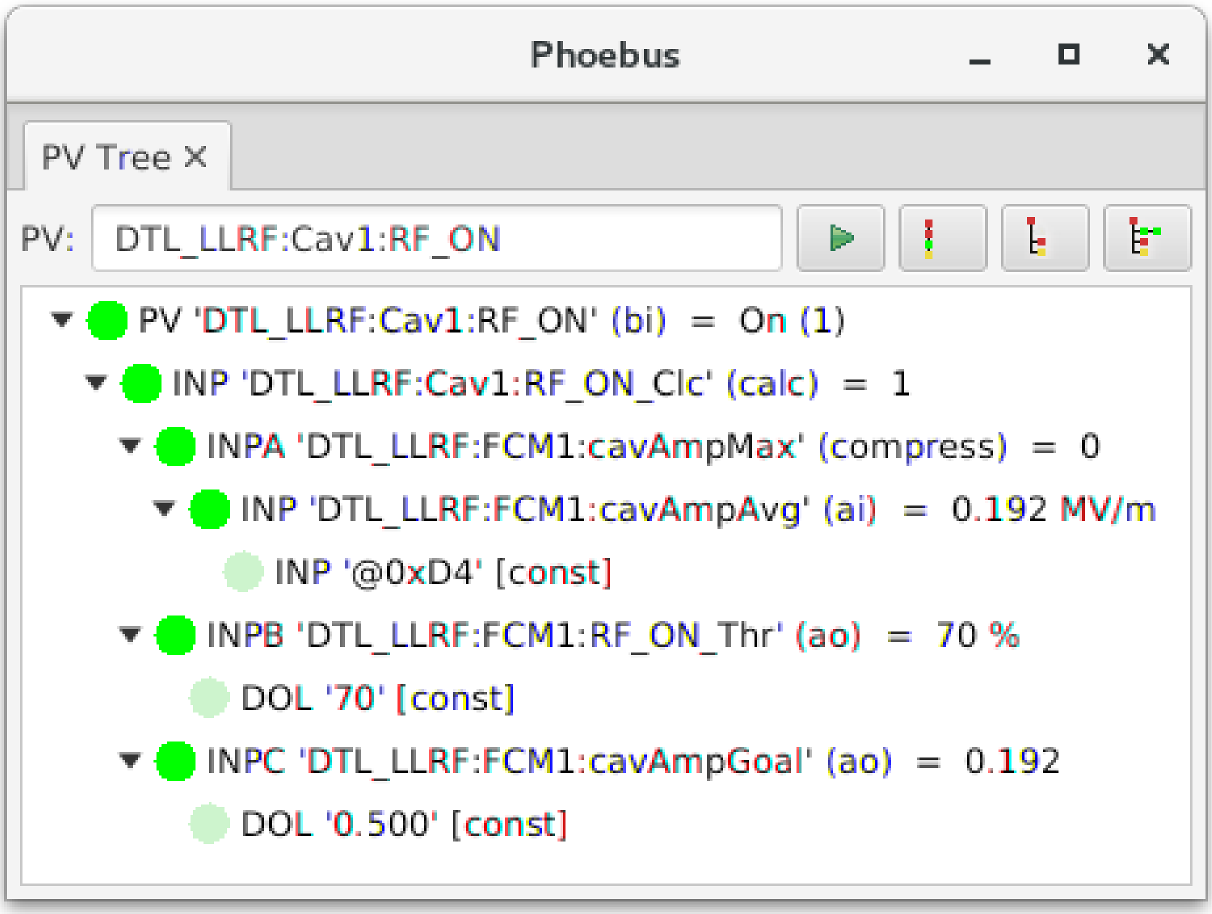
## 3.2 Split Panes within a Window

The tab context menu options to *Split Horizontally* or *Split Vertically* will create sub-panels within the window between which tabs can be arranged.

To un-do a split, simply move all tabs out of a split section. When a split section is empty, it will be merged back with its sibling (unless the pane is named, see below).

## 3.3 Hide Tabs

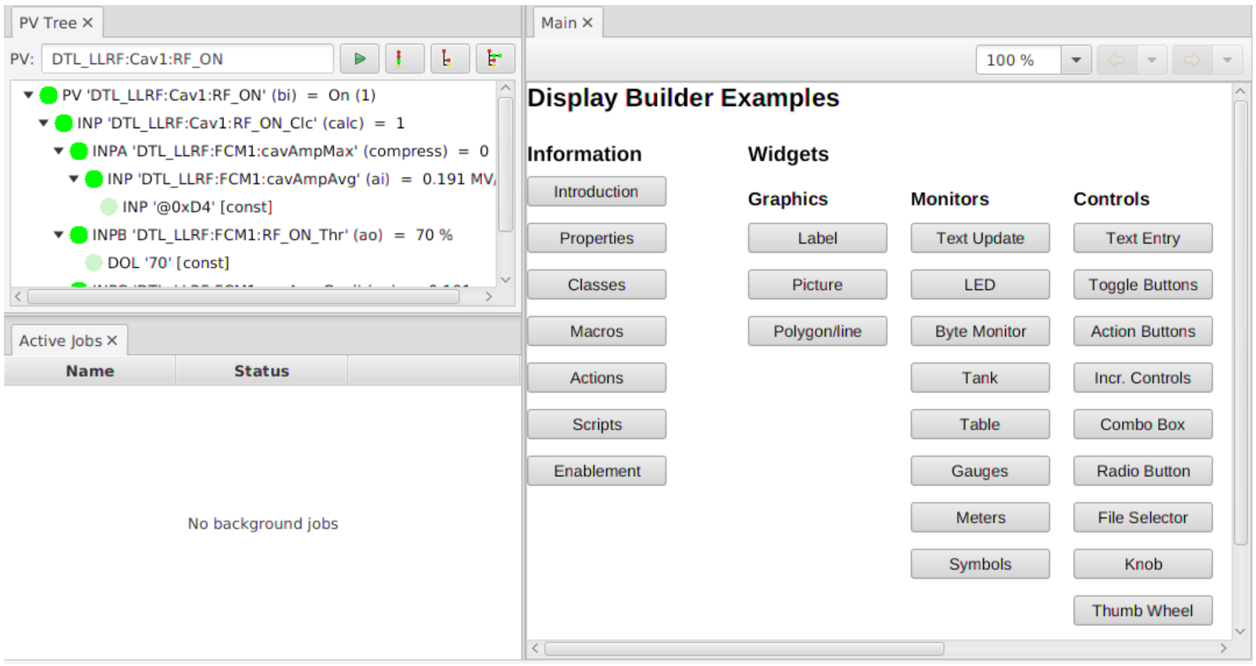
By default, even a window with just one tab will show that tab. This has the advantage that you can then grab that tab to move it into another window, or arrange tabs in the split subsections of a window. At times, however, you may prefer to hide such singular tabs to preserve screen space. In the *Window* menu, select *Always show Tabs* to show respectively hide singular tabs.



PV: DTL_LLRF:Cav1:RF_ON

```
▼ PV 'DTL_LLRF:Cav1:RF_ON' (bi)  =  On (1)
  ▼ INP 'DTL_LLRF:Cav1:RF_ON_Clc' (calc)  =  1
    ▼ INPA 'DTL_LLRF:FCM1:cavAmpMax' (compress)  =  0
      ▼ INP 'DTL_LLRF:FCM1:cavAmpAvg' (ai)  =  0.192 MV/m
          INP '@0xD4' [const]
    ▼ INPB 'DTL_LLRF:FCM1:RF_ON_Thr' (ao)  =  70 %
        DOL '70' [const]
    ▼ INPC 'DTL_LLRF:FCM1:cavAmpGoal' (ao)  =  0.192
        DOL '0.500' [const]
```

Fig. 1: Window with just one tab, showing that tab so you can drag it.



```
PV 'DTL_LLRF:Cav1:RF_ON' (bi)  =  On (1)
  INP 'DTL_LLRF:Cav1:RF_ON_Clc' (calc)  =  1
    INPA 'DTL_LLRF:FCM1:cavAmpMax' (compress)  =  0
      INP 'DTL_LLRF:FCM1:cavAmpAvg' (ai)  =  0.191 MV,
        INP '@0xD4' [const]
    INPB 'DTL_LLRF:FCM1:RF_ON_Thr' (ao)  =  70 %
      DOL '70' [const]
```

No background jobs

Fig. 2: In this example, the original PV Tree tab has been split *horizontally*, creating a left and right section. The left section contains the original PV Tree. A new Display Builder panel has been placed into the originally empty right section. Next, the PV Tree tab on the left has once more been split, this time *vertically*, and a new Jobs viewer has been placed in the newly created bottom half.

## 3.4 Select Tab From Menu

With a large number of open tabs - potentially scattered over multiple application windows - it might be difficult to locate a particular display. In the *Window* menu, the *Select Tab* option will list all tabs identified by title. The list is ordered alphabetically, and the currently selected tab is checked. If tabs have been put in detached windows, multiple tabs will be marked as selected.

Selecting an item in the list will bring that tab to the foreground.

## 3.5 Close All Tabs

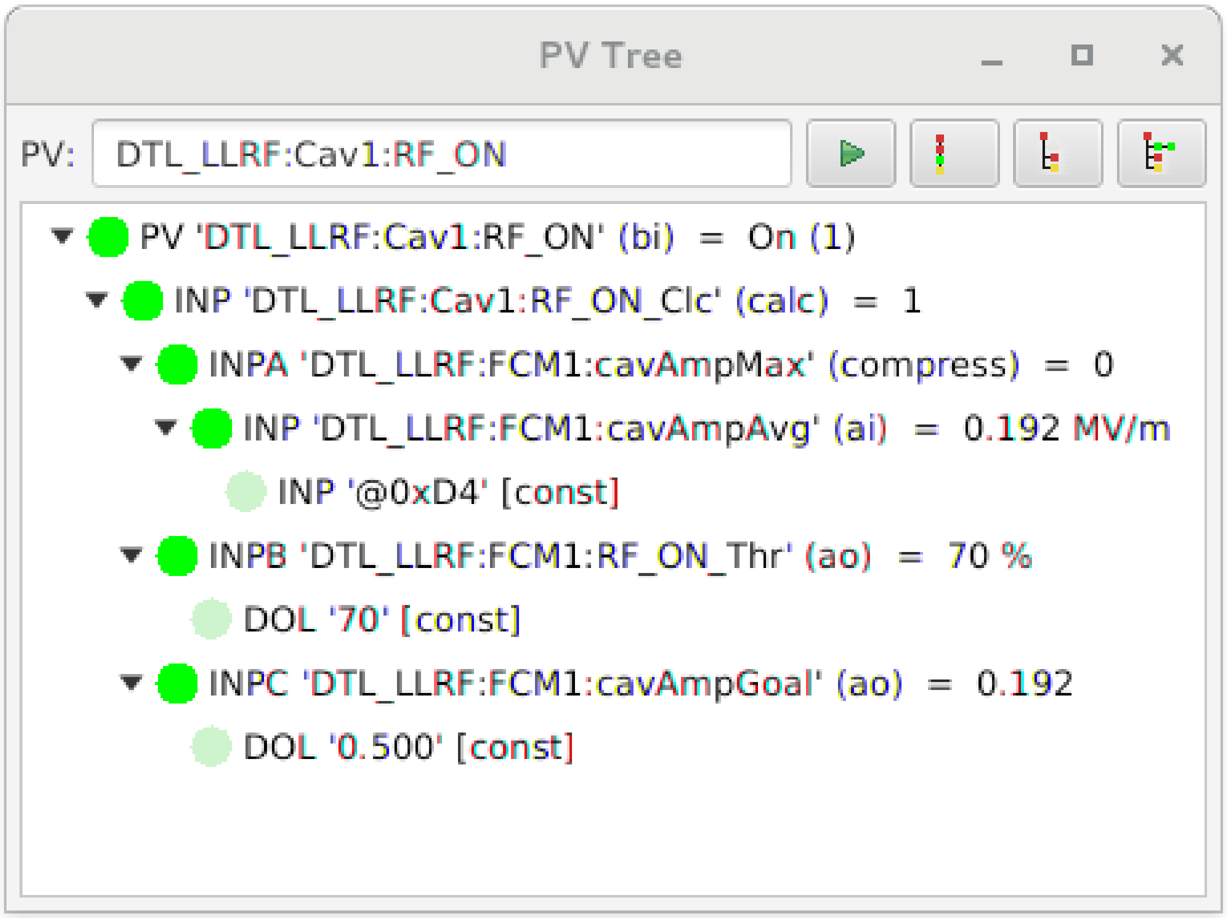
To close all tabs across all windows (main and detached windows), use the *Close All Tabs* menu option from the *Window* menu.

## 3.6 Saving & Restoring the Window Layout

The current window layout is saved to a `memento` file when exiting the program. This `memento` file is by default located within a `.phoebus` subdirectory of the user's home directory. To change the location from `$HOME/.phoebus` to a custom location, set the Java System property `phoebus.user` to the desired location.

When later starting the program back up, it will load the saved window layout.

By making the `memento` file read-only, system administrators can prevent the program from updating the file on exit. Each time the program is started, it will thus start out with a known window layout.

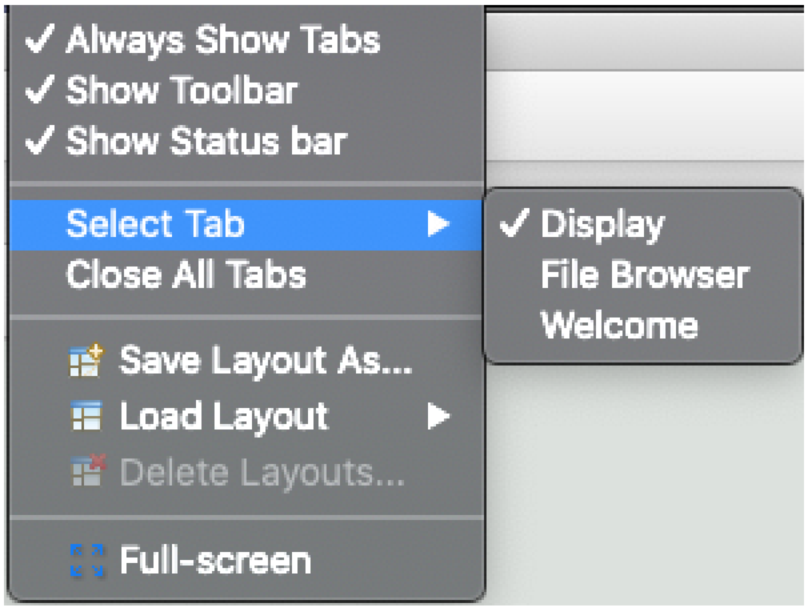


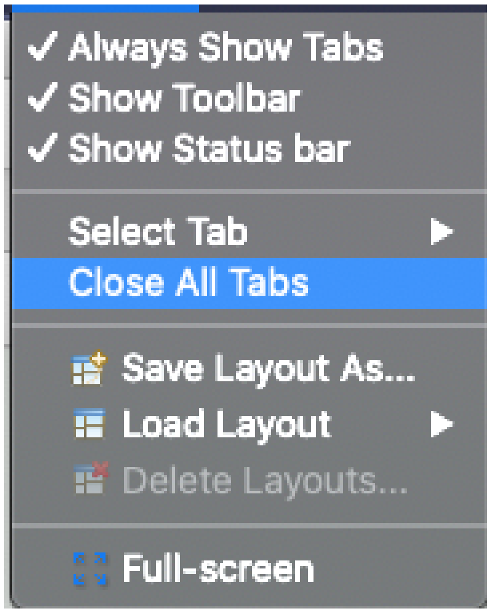
PV Tree

PV: DTL_LLRF:Cav1:RF_ON

```
PV 'DTL_LLRF:Cav1:RF_ON' (bi) = On (1)
  INP 'DTL_LLRF:Cav1:RF_ON_Clc' (calc) = 1
    INPA 'DTL_LLRF:FCM1:cavAmpMax' (compress) = 0
      INP 'DTL_LLRF:FCM1:cavAmpAvg' (ai) = 0.192 MV/m
        INP '@0xD4' [const]
    INPB 'DTL_LLRF:FCM1:RF_ON_Thr' (ao) = 70 %
      DOL '70' [const]
    INPC 'DTL_LLRF:FCM1:cavAmpGoal' (ao) = 0.192
      DOL '0.500' [const]
```

Fig. 3: Window with just one tab, hiding the actual tab to offer more screen space.





The application *Window* menu option *Save Layout As..* allows saving the current window layout under a name. The *Window* menu option *Load Layout* offers a list of all saved layout. Selecting a saved layout switches from the current display layout to a saved one.

Saved, named layouts are stored in files similar to the default `memento`, but including the name of the saved layout. These saved layout memento files can be deleted is no longer needed, copied to different installations, or made read-only to prevent replacement by end users.

## 3.7 Locking the Window Layout

The context menu of a tab within a pane allows locking and un-locking a pane.

A locked pane keeps its current set of tabs. Tabs cannot be moved out of a locked pane, they cannot be closed, nor can new tabs be added to a locked pane. A locked pane cannot be split.

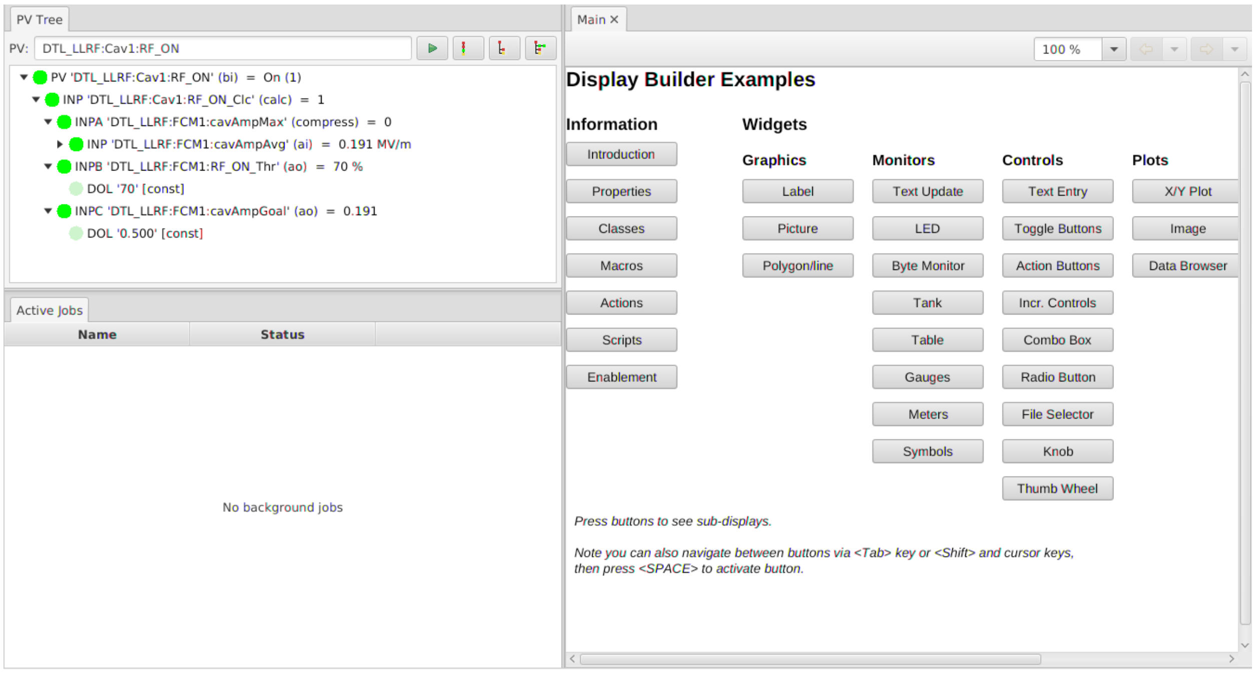
Locked panes allows you to create a default layout that contains certain fixed panes which the user cannot accidentally delete at runtime.

## 3.8 Named Panes

The context menu of a tab allows naming the pane which contains the tab.

Display Builder builder panels can be configured to open new tabs in a specific, named pane. If that pane does not exist, it will be created, but ideally such displays are used within a layout that already contains the appropriately named panes.

A named pane will remain visibly even when empty. It will not be merged with sibling tabs.



```
PV 'DTL_LLRF:Cav1:RF_ON' (bi) = On (1)
  INP 'DTL_LLRF:Cav1:RF_ON_Clc' (calc) = 1
    INPA 'DTL_LLRF:FCM1:cavAmpMax' (compress) = 0
      INP 'DTL_LLRF:FCM1:cavAmpAvg' (ai) = 0.191 MV/m
    INPB 'DTL_LLRF:FCM1:RF_ON_Thr' (ao) = 70 %
      DOL '70' [const]
    INPC 'DTL_LLRF:FCM1:cavAmpGoal' (ao) = 0.191
      DOL '0.500' [const]
```

No background jobs

*Press buttons to see sub-displays.*

*Note you can also navigate between buttons via <Tab> key or <Shift> and cursor keys, then press <SPACE> to activate button.*

Fig. 4: In this example, the upper and lower panes on the left are locked. Note that the tabs for the PV Tree and Active Jobs have no `x` to close them. These tabs cannot be closed, they cannot be moved to other window sections, and you can no longer add new tabs into these *locked* panes.

CHAPTER 4

# Preference Settings

When you run Phoebus, you may find that it cannot connect to your control system because for example the EPICS Channel Access address list is not configured.

To locate available preferences, refer to the complete *Preferences Listing* or check the source code for files named `*preferences.properties`, for example in the `core-pv` sources:

```
# ----------------------------------------
# Package org.phoebus.applications.pvtable
# ----------------------------------------

# Show a "Description" column that reads xxx.DESC?
show_description=true

# -------------------------
# Package org.phoebus.pv.ca
# -------------------------

# Channel Access address list
addr_list=
```

Create a file `settings.ini` that lists the settings you want to change:

```
# Format:
#
#  package_name/setting=value
org.phoebus.pv.ca/addr_list=127.0.0.1 my_ca_gateway.site.org:5066
```

The `value` might be plain text, with details depending on the preference setting, for example allowing an IP address for the `addr_list` or a `true` value for some boolean preference setting. In addition, Java properties or environment variables can be used like this:

```
# Example of using a Java property 'gateway'.
# If it is set to 'my_ca_gateway.site.org:5066',
# this example would have the same end result as
```

(continues on next page)

(continued from previous page)

```
# the previous example.
#
# If no Java property 'gateway' is found,
# an environment variable 'gateway' is checked.
org.phoebus.pv.ca/addr_list=127.0.0.1 $(gateway)
```

Start Phoebus like this to import the settings from your file:

```
phoebus.sh -settings /path/to/settings.ini
```

Start Phoebus like this to import the settings from a remote URL:

```
phoebus.sh -settings http://mysite.com/settings.ini
```

Loading from URL assumes remote service does not respond with a redirect. Moreover, if using https, the remote URL must be backed by a trusted certificate.

At runtime, you can view the currently effective preference settings from the menu `Help`, `About`. The `Details` pane includes a tab that lists all preference settings in the same format that is used by the `settings.ini` file. You can copy settings that you need to change from the display into your settings file.

The same details pane that lists current preference settings also offers an `Import Preferences` button for loading a `settings.ini` file. You may use that as an alternative to the command line `-settings ..` option, but note that settings loaded via this button only become effective after a restart.

Settings loaded via either the `-settings ..` command line option or the `Import Preferences` button are stored in the user location (see *Locations*). They remain effective until different settings are loaded or the user location is deleted. It is therefore not necessary to always run the application with the same `-settings ..` command line option. Just invoking with the command line option once or using the `Import Preferences` button once suffices to load settings. In practice, however, it is advisable to include the `-settings ..` command line option in a site-specific application start script. This way, new users do not need to remember to once start with the option, and existing users will benefit from changes to the settings file.

Conceptually, preference settings are meant to hold critical configuration parameters like the control system network configuration. They are configured by system administrators, and once they are properly adjusted for your site, there is usually no need to change them.

Most important, these are not settings that an end user would need to see and frequently adjust during ordinary use of the application. For such runtime settings, each application needs to offer user interface options like context menus or configuration dialogs.

When you package phoebus for distribution at your site, you can also place a file `settings.ini` in the installation location (see *Locations*). At startup, Phoebus will automatically load the file `settings.ini` from the installation location, eliminating the need for your users to add the `-settings ..` command line option.

## 4.1 Developer Notes

In your code, create a file with a name that ends in `*preferences.properties`. In that file, list the available settings, with explanatory comments:

```
# ----------------------------------------
# Package org.phoebus.applications.my_app
# ----------------------------------------

# Note that the above
```

(continues on next page)

(continued from previous page)

```
#
#    "# Package name.of.your.package"
#
# is important. It is used to format the online help,
# and users will need to know the package name to
# assemble their settings file.

# Explain what each setting means,
# what values are allowed etc.
my_setting=SomeValue

# Enable some feature, allowed values are true or false
my_other_setting=true
```

In your application code, you can most conveniently access them like this:

```
package org.phoebus.applications.my_app

import org.phoebus.framework.preferences.AnnotatedPreferences;
import org.phoebus.framework.preferences.Preference;

class MyAppSettings
{
    @Preference public static String my_setting;
    @Preference public static boolean my_other_setting;

    static
    {
        AnnotatedPreferences.initialize(MyAppSettings.class, "/my_app_preferences.
↪properties");
    }
}
```

The `AnnotatedPreferences` helper will read your `*preferences.properties`, apply updates from `java.util.prefs.Preferences`, and then set the values of all static fields annotated with `@Preference`. It handles basic types like `int`, `long`, `double`, `boolean`, `String`, `File`. It can also parse comma-separated items into `int[]` or `String[]`.

By default, it uses the name of the field as the name of the preference setting, which can be overridden via `@Preference(name="name_of_settings")`. If more elaborate settings need to be handled, `AnnotatedPreferences.initialize` returns a `PreferencesReader`, or you could directly use that lower level API like this:

```
package org.phoebus.applications.my_app

import org.phoebus.framework.preferences.PreferencesReader;

# The class that you pass here determines the package name for your preferences
final PreferencesReader prefs = new PreferencesReader(getClass(), "/my_app_
↪preferences.properties");

String pref1 = prefs.get("my_setting");
Boolean pref2 = prefs.getBoolean("my_other_setting");
// .. use getInt, getDouble as needed.
// For more complex settings, use `get()` to fetch the string
// and parse as desired.
```

The `PreferencesReader` loads defaults from the property file, then allows overrides via the `java.util.prefs.Preferences` API. By default, the user settings are stored in a `.phoebus` folder in the home directory. This location can be changed by setting the Java property `phoebus.user`.

In the future, a preference UI might be added, but as mentioned the preference settings are not meant to be adjusted by end users.

CHAPTER 5

# Logging

All phoebus code logs via the `java.util.logging` mechanism.

The default log settings for the phoebus product are based on the `logging.properties` file of the `core-launcher` module, which can be downloaded from https://github.com/ControlSystemStudio/phoebus/blob/master/core/launcher/src/main/resources/logging.properties. Services like the alarm server have a similar built-in log configuration file, for instance https://github.com/ControlSystemStudio/phoebus/blob/master/services/alarm-server/src/main/resources/alarm_server_logging.properties.

At runtime, the log settings of the product can be adjusted via the "Logging Configuration" application, which is most convenient for one-time changes. To adjust the log settings of the product more permanently, or to adjust the log settings of services which do not have a GUI, you can use a command line option to override the built-in logging properties. Create a copy of the file, adjust as desired, and pass it via the `-logging /path/to/my_logging.properties` command line option.

The default configuration sends log messages to the console, i.e. the terminal window from which the product was started. On Windows, there might not be a terminal, and on other systems, a launcher script might redirect the console output. The "Error Log" application allows viewing log messages in the product GUI.

# CHAPTER 6

# Runtime Settings

When you run Phoebus, you may open a tab with the "PV Tree", enter a PV name, move the window around etc.

When you exit Phoebus, the current location of windows, tabs, and for example the PV name of an active "PV Tree" are stored.

When you later restart Phoebus, it restores the windows with their saved location and content.

## 6.1 Developer Notes

The state is persisted in a `memento` file, located in the same directory as the user preferences. To change the default from `.phoebus/memento` in your home directory to a different location, see Preference Settings *Developer Notes*.

To always start Phoebus with a known window layout, save the `memento` from a desired layout, and then place that saved file in the user preferences directory before starting a new instance of Phoebus.

CHAPTER 7

# Authorization

## 7.1 Authentication vs Authorization

Phoebus depends on the operating system to authenticate the user. The currently logged in user is who we expect to be interacting with Phoebus.

Phoebus does add basic authorization to control if the current user may configure details of the window layout (lock and unlock panes) or alarm system (add, remove, acknowledge alarms).

## 7.2 Configuring Authorization

A preference setting selects a more detailed authorization configuration file:

```
org.phoebus.ui/authorization_file=/path/to/authorization.conf
```

See details of the `org.phoebus.ui` preferences for the possible locations of that file.

Example authorization configuration file:

```
# Authorization Settings
#
# Format:
# authorization = Comma-separated list of user names
#
# The authorization name describes the authorization.
# FULL is a special name to obtain all authorizations.
#
# Each entry in the list of user names is a regular expression for a user name.

# Anybody can lock a dock pane, i.e. set it to 'fixed'
lock_ui = .*

```

(continues on next page)

(continued from previous page)

```
# Anybody can acknowledge alarms
alarm_ack = .*

# Specific users may configure alarms, including both "jane" and "janet"
#alarm_config = fred, jane.*, egon,

# Anybody can configure alarms
alarm_config = .*

# Full authorization.
FULL = root
```

# CHAPTER 8

# Applications

The following sections describe details of specific application features.

## 8.1 Core Framework

The core framework module of phoebus consists of a set of commonly used services, interfaces, and other utilities.

### 8.1.1 Selection Service

The Selection Service provides a clean and powerful mechanism for allowing Phoebus applications to share the user selected data. The use of Selection Service helps to build an integrated set of applications, capable of responding to user operations in other applications, while still avoiding direct dependencies between applications and supporting a modular product.

The selection service allows applications to

1. Publish their selection
2. Register listeners to be notified of changes to the current selection

### 8.1.2 Adapter Service

The Adapter services provided by the framework is a means for Phoebus applications to provide runtime integration between loosely coupled applications.

For example, most Phoebus applications needs to support making logbook entries. Without adapters, each application would have to add a dependency to each of the supported logbook application. However, with the use of adapters each application simply registers an AdapterFactory which describes how its model/selection can be adapted into an object that can be used to create logbook entries. Additionally, by separating the AdapterFactory into a different module, different adapters can be used by different users, each of which support different logbooks. This would not be possible without the use of adapters due to the direct dependencies that would exist between the applications.

Here is an example where an application is including a context menu item to make log entires based on the selection.

case 1.

Without the use of Adapters and Selection service, each potential menu contribution has to be manually included/excluded.

```
ContextMenu menu = new ContextMenu();
if (LogbookPreferences.is_supported)
    items.add(new SendLogbookAction(...));
else if (LogbookPreferences.is_olog_supported)
    items.add(new SendOlogLogbookAction(...));
else if (LogbookPreferences.is_elog_supported)
    items.add(new SendElogLogbookAction(...));
```

And the application would have an explicit dependency on module not necessarily needed.

```
<dependency>
  <artifactId>logbook-ui</artifactId>
   ...
</dependency>
<dependency>
  <artifactId>olog-logbook-ui</artifactId>
   ...
</dependency>
<dependency>
  <artifactId>elog-logbook-ui</artifactId>
   ...
</dependency>
```

Case 2.

With the use of adapters and the selection service, each application simply needs to publish its selection object and register AdapterFactories. Supported items are then include at runtime and the application doe not have hard dependencies to any particular implementation.

```
ContextMenu menu = new ContextMenu();
SelectionService.getInstance().setSelection(this, Arrays.
↪asList(AppSelection(appModel)));
List<ContextMenuEntry> supported = ContextMenuService.getInstance().
↪listSupportedContextMenuEntries();
supported.stream().forEach(action -> {
        MenuItem menuItem = new MenuItem(action.getName(), new ImageView(action.
↪getIcon()));
        ...
        items.add(menuItem);
    });
```

Register zero or more AdapterFactories which provide the mechanism to adapt an AppSelection to a simple LogEntry or an Olog LogEntry

## 8.2 Active Jobs

Opened from the `Applications`, `Debug` menu, the Active Jobs panel displays ongoing background jobs.

Examples include jobs that load configuration files. Long-running jobs can be cancelled.

### 8.2.1 Implementation Detail

Canceling a job asks the application to abort an ongoing job and depends on a proper implementation of the application feature to honor the request.

## 8.3 PV List

Opened from the `Applications`, `Debug` menu, the PV List panel lists all active PVs, their connection state and the reference count.

## 8.4 Credentials Management

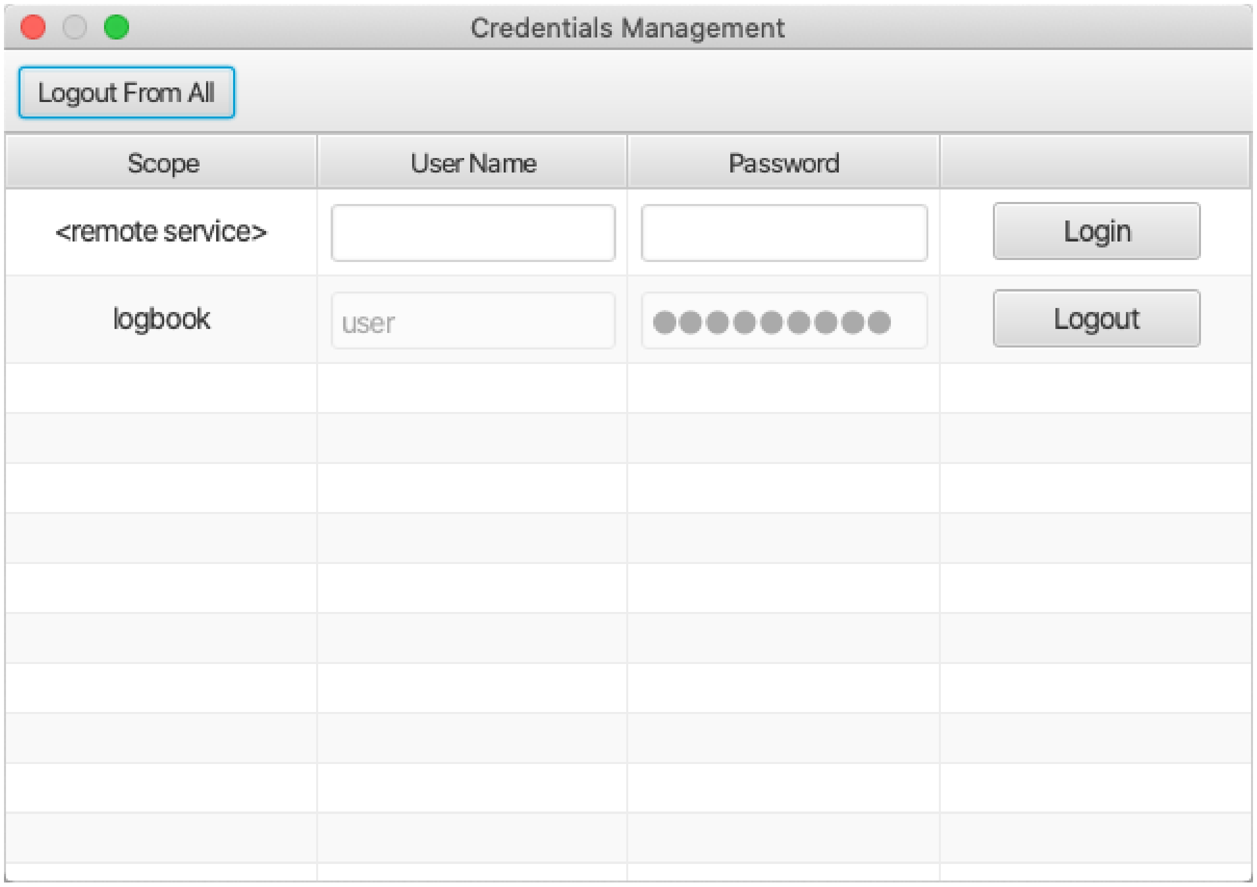
Some applications may need to prompt the user for credentials, e.g. when interacting with a protected remote service. One such example would be the electronic logbook.

Further, Phoebus may be configured to store the credentials entered by the user to avoid repeated prompts. In order to also support an explicit logout capability, the Credentials Management application offers means to remove stored credentials.

In some cases an explicit login procedure can be useful, e.g. login to service for the purpose of storing user credentials and thereby support automated creation of logbook entries.

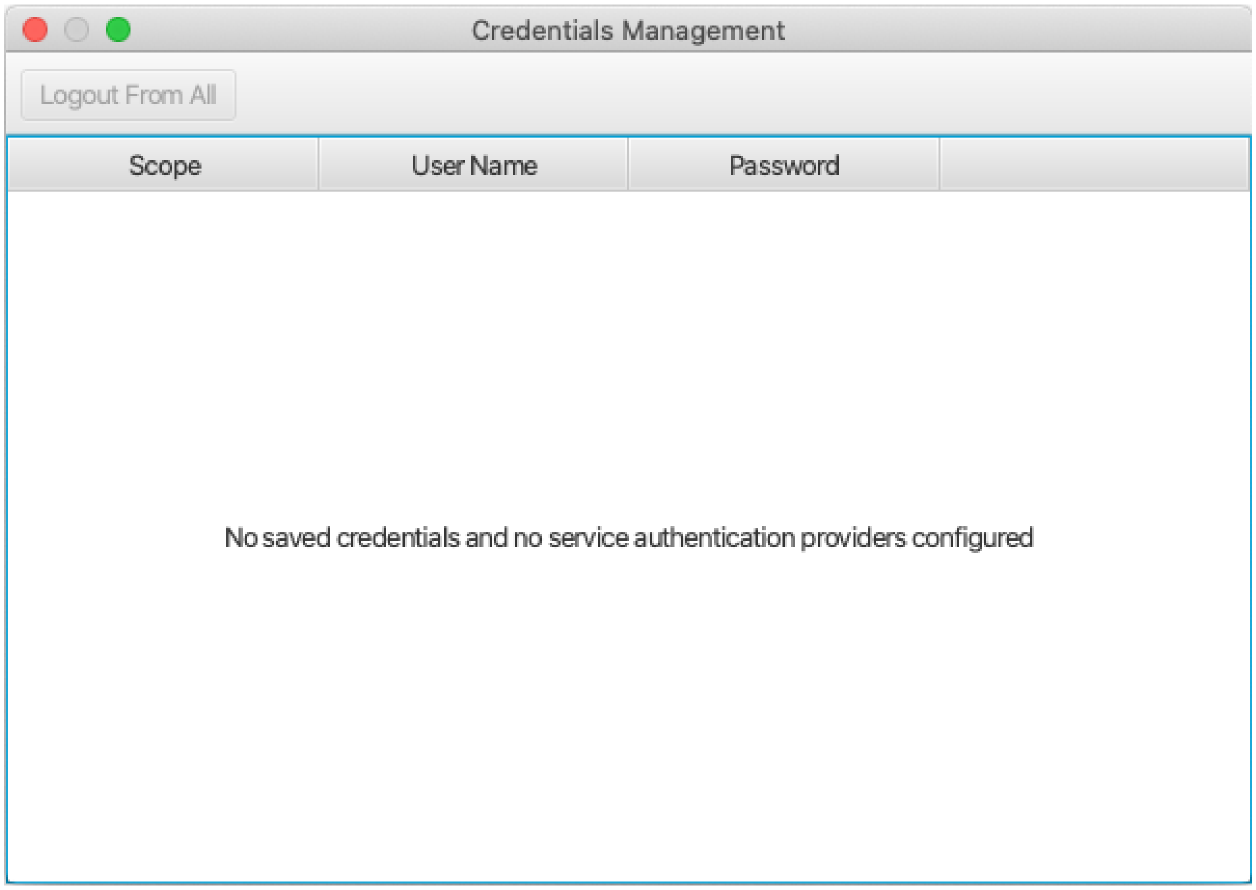
The application is launched using the dedicated button in the (bottom) status bar.

The below screen shot shows an example where credentials have been stored for the "logbook" scope, plus an option to login to the "<remote service>" scope. User may also choose to "logout" from all scopes, i.e. to remove all stored credentials.



| Scope | User Name | Password | |
|---|---|---|---|
| <remote service> | | | Login |
| logbook | user | ●●●●●●●●● | Logout |

If no credentials are stored in the credentials store, and if no services supporting authentication have been configured, the Credentials Management UI will show a static message:



No saved credentials and no service authentication providers configured

## 8.5 Formulas

### 8.5.1 Function categories

*Arrays*

**arrayDiv(VNumberArray array1, VNumberArray array2)** - Returns a VNumberarray where each element is defined as array1[x] / array2[x]. The input arrays must be of equal length.

**arrayDivScalar(VNumberArray array, VNumber denominator)** - Returns a VNumberArray where each element is defined as array[x] / denominator.

**arrayDivScalarInv(VNumber nominator, VNumberArray array)** - Returns a VNumberArray where each element is defined as nominator / array[x].

**arrayMult(VNumberArray array1, VNumberArray array2)** - Returns a VNumberArray where each element is defined as array1[x] * array2[x]. The input arrays must be of equal length.

**arrayOf([VString | VNumber]. . . )** - Returns a VStringArray or VNumberArray depending on the input argument types. All elements of the input argument list must be either VString or VNumber.

**arrayPow(VNumberArray array, VNumber exponent)** - Returns a VNumberArray where each element is defined as pow(array[x], exponent).

**arraySum(VNumberArray array, VNumber offset)** - Returns a VNumberArray where each element is defined as array[x] + offset. To subtract, use negative offset.

**elementAt([VNumberArray | VStringArray] array , VNumber index)** - Returns a VNumber or VString at the specified index of the input array. If the index is invalid, a NaN or empty string is returned.

**histogramOf(VNumberArray array [, VNumber binCount])** - Computes a histogram for the input array. The binCount argument is optional and defaults to 100.

**scale(VNumberArray array, VNumber factor [,VNumber offset])** - Returns a VNumberArray where each element is defined as array[x] * factor [+ offset]. The offset is optional.

**subArray([VNumberArray | VStringArray] array, VNumber fromIndex, VNumber toIndex)** - Returns a VNumberArray or VStringArray that is a sub-array defined by the fromIndex and toIndex. The indexes must be valid, e.g. fromIndex > 0, fromIndex < toIndex etc.

**arrayRangeOf(VNumberArray array)** - Returns a Display Range of the given array This includes the display min, max

**arrayStats(VNumberArray array)** - Returns a VStatistic with the statistical information of the given array This includes the average, min, max, and element count

**arrayMax(VNumberArray array)** - Returns a VDouble with the greatest value of the given array

**arrayMin(VNumberArray array)** - Returns a VDouble with the smallest value of the given array

## 8.6 Process Variables

Several types of process variables are supported. A prefix of the form "xx://.." is typically used to select the PV type.

### 8.6.1 Channel Access

Process variables that are to be accessed over the channel access protocol are simply identified by the channel name.

Channel access is the default protocol. If desired, 'ca://' can be used to specifically select channel access, but for the time being no protocol identification is necessary for channel access.

Examples:

```
SomePVName
ca://SomePVName
SomeMotor.RBV
```

Channel Access settings are configured via *Preferences Listing*, most important:

```
# Channel Access address list
org.phoebus.pv.ca/addr_list=...
```

### 8.6.2 PV Access

Process variables that are to be accessed over the PV Access protocol must be identified by a formatted string that contains the process variable's name.

As PV Access is not the default protocol, process variables accessed over it must have the protocol spectrue;ified with 'pva://'.

The PV Access format is as follows:

```
pva://SomePVName
pva://SomePVName/subfield/subelement
```

As shown, when accessing structures, the path to a nested structure element can be provided.

PV Access is configured via the following environment variables or Java properties:

```
# Address list. When empty, local subnet is used
export EPICS_PVA_ADDR_LIST = "127.0.0.1  10.1.10.255"

# Add local broadcast addresses to addr list? (Value YES, NO)
export EPICS_PVA_AUTO_ADDR_LIST = YES
```

### 8.6.3 Simulated

Simulated process variables are useful for tests. They do not communicate with the control system.

**The provided simulated process variables are:**

- flipflop(update_seconds)
- gaussianNoise(center, std_dev, update_seconds)
- gaussianWave(period, std_dev, size, update_seconds)
- intermittent(update_seconds)
- noise(min, max, update_seconds)
- noisewave(min, max, update_seconds)
- ramp(min, max, update_seconds)
- sawtooth(period_seconds, wavelength, size, update_seconds)
- sine(min, max, update_seconds)
- sinewave(period_seconds, wavelength, size, update_seconds)
- strings(update_seconds)
- const(value)

Examples:

```
sim://sine
sim://ramp
sim://ramp(1, 10, 0.2)
sim://noise
sim://const(42)
sim://const("Fred")
```

### 8.6.4 Local

Local process variables can be used within the application, for example to send a value from one display to another display within the same application. They do not communicate with the control system.

Following the “loc://” prefix, the variable name must start with a character A-Z or a-z, potentially followed by more characters or numbers. Valid examples are “A”, “b”, “Example2”, “MyVar42”. Invalid examples are “42”, “2ndExample”.

Next is an optional type selector like "<VLong>" and initial value like "42". Unless a type selector and initial value are provided, a local value will be of type 'VDouble' with initial value of 0.

Local process variables only exist as long as they are referenced. When all references to a local process variable are released, the PV is deleted.

Examples:

```
loc://example
loc://a_number(42.2)
loc://an_array(3.8, 2.5, 7.4)
loc://a_text("Hello")
loc://large<VLong>(42)
loc://options<VEnum>(2, "A", "Initial", "B", "C")
```

### 8.6.5 Formulas

Formula-based PVs can perform simple computations on constants and other PVs. The equation can be created via the 'eq://' prefix or alternatively via '='. Other process variables are referenced via backquotes.

Examples:

```
eq://3+4
=3+4
=10 + 5*`sim://sine`
=`sim://ramp`>1 ? 10 : -10
```

### 8.6.6 MQTT

Data that is to be read over the MQTT network protocol must be referenced with a formatted string which contains the name of the MQTT topic and the VType that corresponds to the type of data published on the topic.

All MQTT topics are obtained from the same MQTT broker URL, based on a preference setting that defaults to:

```
org.phoebus.pv.mqtt/mqtt_broker=tcp://localhost:1883
```

If the VType is omitted, 'double' is assumed. Examples:

```
mqtt://some_topic
mqtt://some_topic<VDouble>
mqtt://some_topic<VString>
mqtt://some/nested/topic
```

### 8.6.7 System

System process variables are useful for representing some system attributes. They do not communicate with the control system.:

```
* sys://time
* sys://timeOffset(offset, format, update_seconds)
```

The *timeOffset* pv allows you to represent a time instant offset from *now*. The optional parameters are: *offset* which is described as 1 min, 1 hour, 1 day prior to the current instant. *format* which describes how to represent the instance, the supported formats are full, milli, seconds, datetime, date, or time. *update_seconds* the update rate / period.

Examples

```
sys://timeOffset(12 hours)
sys://timeOffset(1hour, time, 1)
```

### 8.6.8 Tango

Tango is different from EPICS, the smallest unit is Device, which includes the commands, states, and attributes. The command and the attribute has been implemented, add prefix to PV Name in editing interface to distinguish, and the command usually has a return value, so need to use *Text Entry* or a combination of *Action button* and *Text Update* components to achieve it. Currently, all types of scalars are supported, but SPECTRUM and IMAGE are not yet supported.

Examples

```
tga://device/attribute
tgc://device/command
```

## 8.7 Performance Monitor

Invoking the menu Applications, Debug, Performance monitor adds a button to the status bar that displays performance information. Invoking it again will remove it.

### 8.7.1 Memory

Displays how much memory is allocated, and how much of that is available. For example, "Avail: 28% of 0.21GB".

Pressing the button triggers a garbage collection.

### 8.7.2 Frame Rate

Measures the JavaFX frame rate, nominally 60Hz.

System properties that can influence the result:

*-Dquantum.multithreaded=true -Djavafx.animation.fullspeed=true -Djavafx.animation.framerate=120 -Djavafx.animation.pulse=120*

## 8.8 Save-And-Restore

### 8.8.1 Overview

The save-and-restore application can be used to take "snapshots" of a pre-defined list if PVs at a certain point in time, and write the persisted values back at some later point.

The application uses the save-and-restore service deployed on the network such that it can be accessed over HTTP(s). The URL of the service is specified in the save-and-restore.properties file, or in the settings file pointed to on the command line.

### 8.8.2 Nodes and node types

Save-and-restore data managed by the service is arranged in a tree structure and hence presented in the client UI using a tree view UI component. In the following objects in the tree are referred to as "nodes".

The root of the tree structure is a folder that may only contain folder nodes. Folders may contain sub-folders and configurationsh. The child nodes of a configuration are snapshots associated with that configuration.

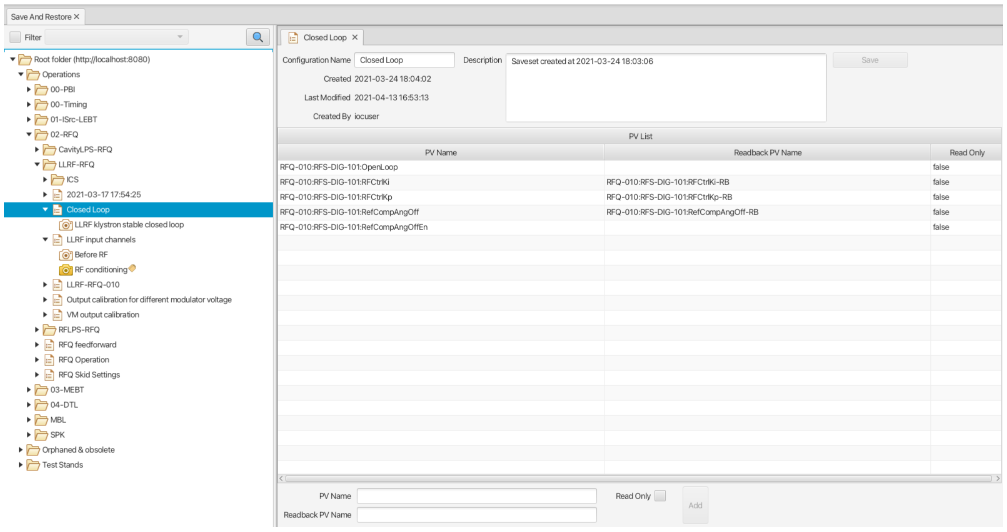
Additionally a composite snapshot node may reference an arbitrary number of snapshot or composite snapshot nodes.

There are thus four node types managed in the application:

- **Folder**: container for folders and configurations.
- **Configuration**: a list of PV names and associated meta-data.
- **Snapshot**: PV values read from PVs listed in a configuration.
- **Composite Snapshot**: aggregation of snapshots or other composite snapshots, or both.

*NOTE*: If a folder or configuration node is deleted, all child nodes are unconditionally and recursively deleted. The user is prompted to confirm delete actions as they are irreversible.

Below screen shot shows the tree structure and a configuration editor.



Configuration Name: Closed Loop
Description: Saveset created at 2021-03-24 18:03:06
Created: 2021-03-24 18:04:02
Last Modified: 2021-04-13 16:53:13
Created By: iocuser

PV List

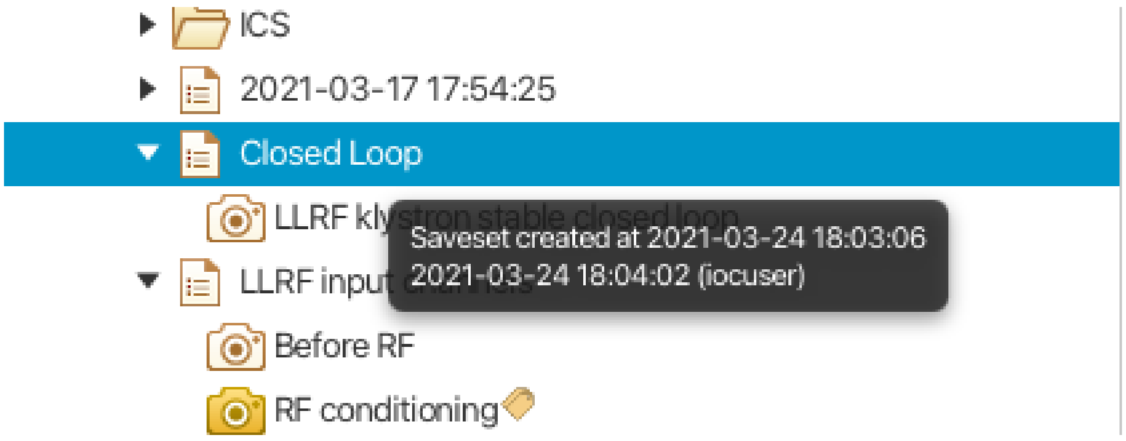
| PV Name | Readback PV Name | Read Only |
|---|---|---|
| RFQ-010:RFS-DIG-101:OpenLoop | | false |
| RFQ-010:RFS-DIG-101:RFCtrlKi | RFQ-010:RFS-DIG-101:RFCtrlKi-RB | false |
| RFQ-010:RFS-DIG-101:RFCtrlKp | RFQ-010:RFS-DIG-101:RFCtrlKp-RB | false |
| RFQ-010:RFS-DIG-101:RefCompAngOff | RFQ-010:RFS-DIG-101:RefCompAngOff-RB | false |
| RFQ-010:RFS-DIG-101:RefCompAngOffEn | | false |

PV Name:
Readback PV Name:
Read Only: ☐

### 8.8.3 Node names and ordering

Node names are case sensitive. Within a parent node child node names must be unique between nodes of same type.

Child nodes in the tree view are ordered first by type (folders, configurations, composite snapshots), then by name. Child nodes of configurations can only be of type snapshot, so these are ordered by name. The tooltip of a node will provide information on date created and user name:



Saveset created at 2021-03-24 18:03:06
2021-03-24 18:04:02 (iocuser)

### 8.8.4 A word of caution

Save-and-restore data is persisted in a central service and is therefore accessible by multiple clients. Users should keep in mind that changes (e.g. new or deleted nodes) are not pushed to all clients. Caution is therefore advocated when working on the nodes in the tree, in particular when changing the structure by deleting or moving nodes.

### 8.8.5 Drag-n-drop

Nodes in the tree can be copied (mouse + modifier key) or moved using drag-n-drop. The following restrictions apply: * Only folder and configuration nodes can be copied or moved. * Configuration nodes cannot be copied or moved to the root folder node. * Target node (i.e. drop target) must be a folder.

Checks are performed on the service to enforce the above restrictions. If pre-conditions are not met when the selection is dropped, the application will present an error dialog.

Drag-n-drop is disabled if multiple nodes are selected and if: * Selection contains a combination of folder and configuration nodes. Selected nodes must be of same type. * Selection contains nodes with different parent nodes. Selected nodes must have the same parent node.

Once a selection of nodes have been copied or moved successfully, the target folder is refreshed to reflect the change.

**NOTE**: Copying a large number of nodes and/or nodes with deep sub-trees is discouraged as this is an "expensive" operation. Moving nodes on the other hand is lightweight as only references in the tree structure are updated.

### 8.8.6 Logging

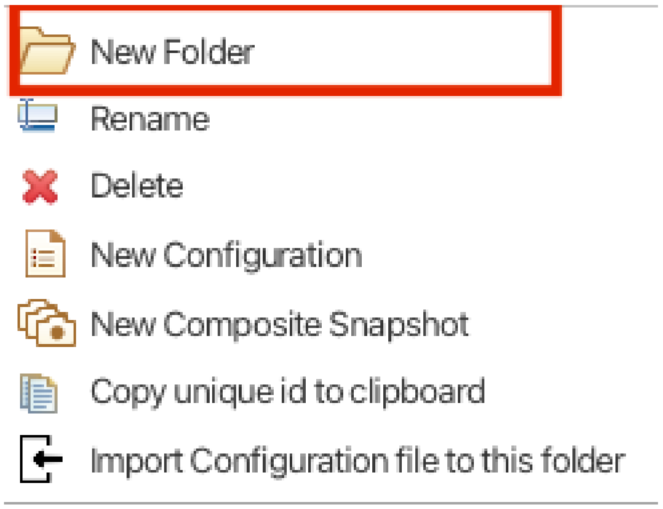
If a logbook implementation is available in the application, the optional logging module can be used to launch a log entry editor for the purpose of logging when a new snapshot has been saved or restored. Properties of the snapshot (name, date etc) are automatically set on the log entry rendered by the editor. If a restore action has failed to write one or multiple PVs, a list of these PVs is also added to the log entry.

### 8.8.7 Workflow

The following sections describe typical use cases when working with configurations and snapshots.

### 8.8.8 Folder

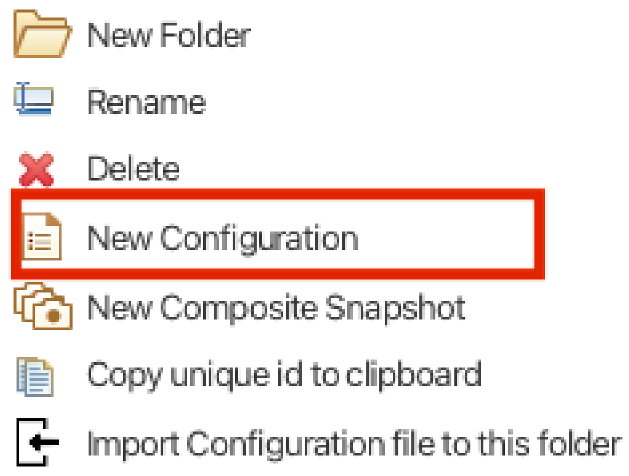
Folder nodes can be created from the New Folder option of the folder node context menu:


New Folder
Rename
Delete
New Configuration
New Composite Snapshot
Copy unique id to clipboard
Import Configuration file to this folder

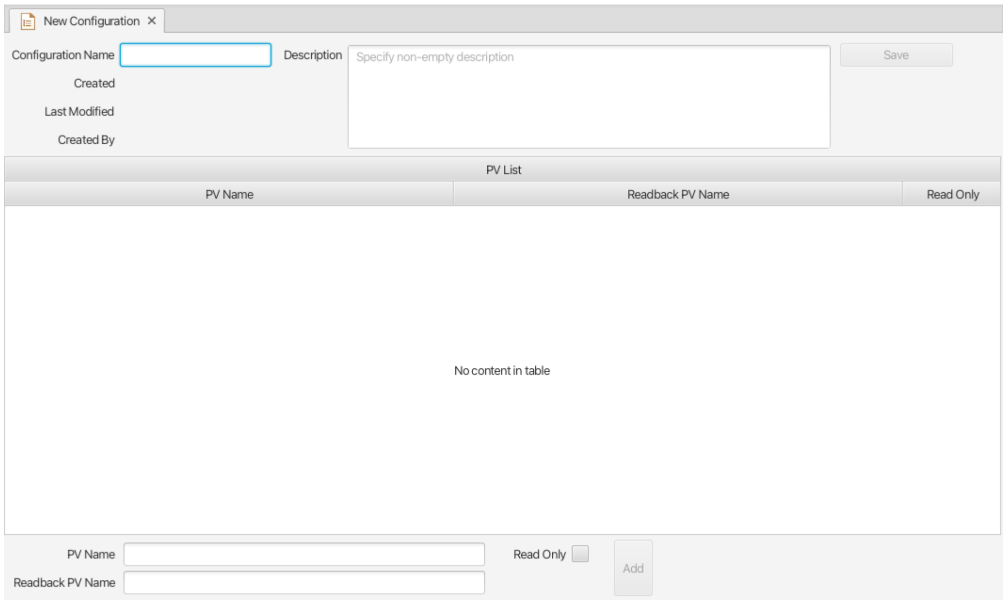
Folder names are case-sensitive and must be unique within the same parent folder.

### 8.8.9 Configuration View

A new configuration is created from the context menu launched when right-clicking on a folder node in the tree view:


New Folder
Rename
Delete
New Configuration
New Composite Snapshot
Copy unique id to clipboard
Import Configuration file to this folder

This will launch the configuration editor:

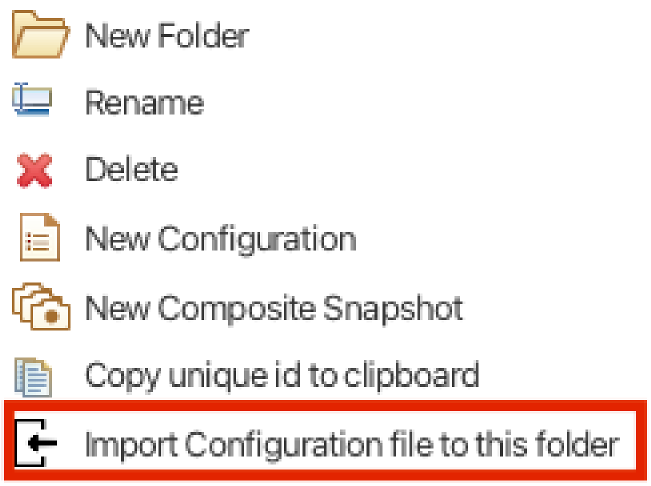


PVs are added using the input field at the bottom of the view. Multiple PV names may be added if separated by space or semicolon.

Specifying a read-back PV is optional. Specifying multiple read-back PV names is supported, and these will be associated with PVs in the order they appear.

PV entries in a configuration marked as read only will be omitted whe performing a restore operation.

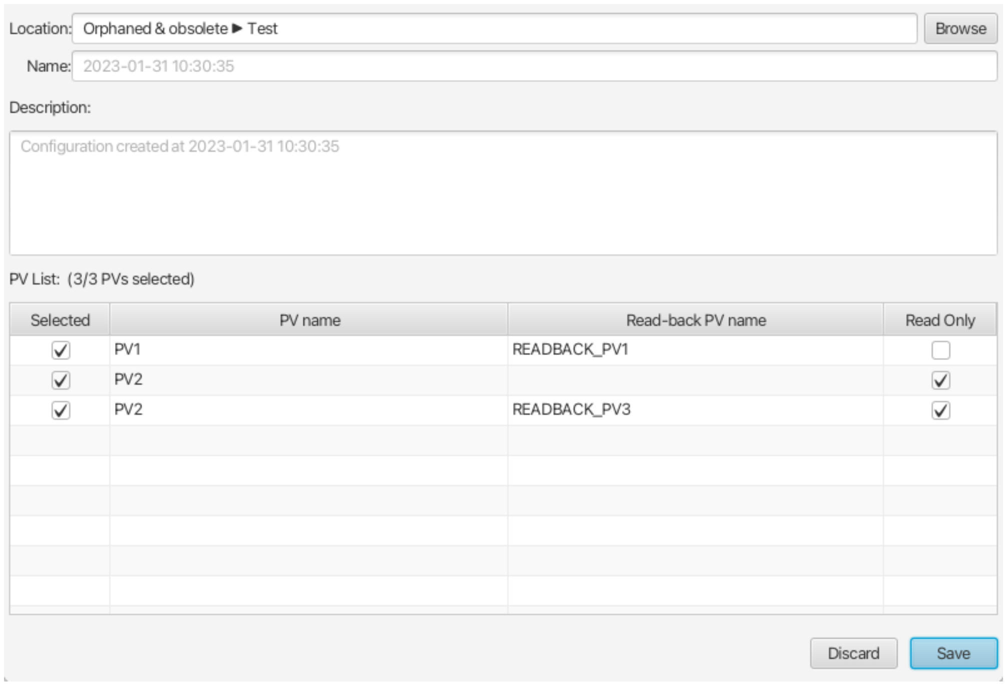
To add a very large number of PVs, user should consider the import feature available via the “Import Configuration file to this folder” option in the context menu of a folder node in the tree view:


New Folder
Rename
Delete
New Configuration
New Composite Snapshot
Copy unique id to clipboard
Import Configuration file to this folder

The file format for such a file is:

```
PV,READBACK,READ_ONLY
PV1,READBACK_PV1,0
PV2,,1
PV2,READBACK_PV3,1
.
.
.
```

The first line is a heading an cannot be omitted. An import of the above example will launch a dialog like so:



Location: Orphaned & obsolete ▶ Test

Name: 2023-01-31 10:30:35

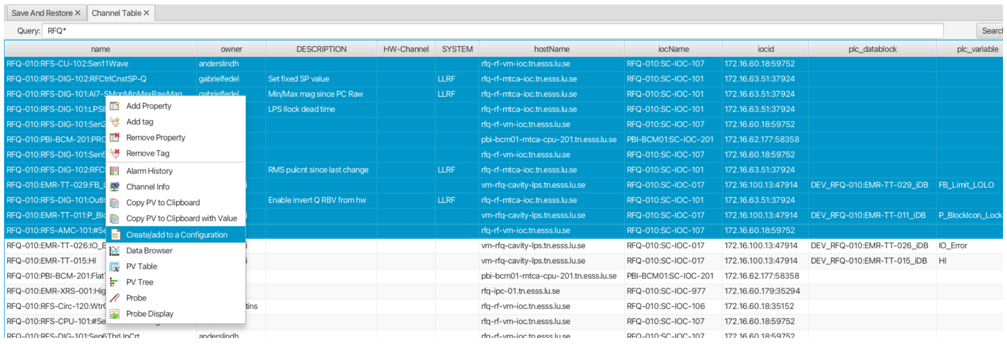
Description:

Configuration created at 2023-01-31 10:30:35

PV List: (3/3 PVs selected)

| Selected | PV name | Read-back PV name | Read Only |
|---|---|---|---|
| ✓ | PV1 | READBACK_PV1 | ☐ |
| ✓ | PV2 | | ✓ |
| ✓ | PV2 | READBACK_PV3 | ✓ |

Another option to add a list of PVs is to use the Channel Table application. In the table user may select wanted PVs and launch the context menu to create a new configuration, or to add the selected PVs to an existing configuration:



Query: RFQ*

| name | owner | DESCRIPTION | HW-Channel | SYSTEM | hostName | iocName | iocid | plc_datablock | plc_variable |
|---|---|---|---|---|---|---|---|---|---|
| RFQ-010:RFS-CU-102:Sen11Wave | anderslindh | | | | rfq-rf-vm-ioc.tn.esss.lu.se | RFQ-010:SC-IOC-107 | 172.16.60.18:59752 | | |
| RFQ-010:RFS-DIG-102:RFCtrlCnstSP-Q | gabrielfedel | Set fixed SP value | | LLRF | rfq-rf-mtca-ioc.tn.esss.lu.se | RFQ-010:SC-IOC-101 | 172.16.63.51:37924 | | |
| RFQ-010:RFS-DIG-101:AI7-SMonMinMaxRawMag | gabrielfedel | Min/Max mag since PC Raw | | LLRF | rfq-rf-mtca-ioc.tn.esss.lu.se | RFQ-010:SC-IOC-101 | 172.16.63.51:37924 | | |
| RFQ-010:RFS-DIG-101:LPSI | | LPS Ilock dead time | | | rfq-rf-mtca-ioc.tn.esss.lu.se | RFQ-010:SC-IOC-101 | 172.16.63.51:37924 | | |
| RFQ-010:RFS-DIG-101:Sen2 | | | | | rfq-rf-vm-ioc.tn.esss.lu.se | RFQ-010:SC-IOC-107 | 172.16.60.18:59752 | | |
| RFQ-010:PBI-BCM-201:PRC | | | | | pbi-bcm01-mtca-cpu-201.tn.esss.lu.se | PBI-BCM01:SC-IOC-201 | 172.16.62.177:58358 | | |
| RFQ-010:RFS-DIG-101:Sen5 | | | | | rfq-rf-vm-ioc.tn.esss.lu.se | RFQ-010:SC-IOC-107 | 172.16.60.18:59752 | | |
| RFQ-010:RFS-DIG-102:RFC | | RMS pulcnt since last change | | LLRF | rfq-rf-mtca-ioc.tn.esss.lu.se | RFQ-010:SC-IOC-101 | 172.16.63.51:37924 | | |
| RFQ-010:EMR-TT-029:FB_L | | | | | vm-rfq-cavity-lps.tn.esss.lu.se | RFQ-010:SC-IOC-017 | 172.16.100.13:47914 | DEV_RFQ-010:EMR-TT-029_iDB | FB_Limit_LOLO |
| RFQ-010:RFS-DIG-101:Outl | | Enable invert Q RBV from hw | | LLRF | rfq-rf-mtca-ioc.tn.esss.lu.se | RFQ-010:SC-IOC-101 | 172.16.63.51:37924 | | |
| RFQ-010:EMR-TT-011:P_Bl | | | | | vm-rfq-cavity-lps.tn.esss.lu.se | RFQ-010:SC-IOC-017 | 172.16.100.13:47914 | DEV_RFQ-010:EMR-TT-011_iDB | P_BlockIcon_Lock |
| RFQ-010:RFS-AMC-101:#Se | | | | | rfq-rf-vm-ioc.tn.esss.lu.se | RFQ-010:SC-IOC-107 | 172.16.60.18:59752 | | |
| RFQ-010:EMR-TT-026:IO_E | | | | | vm-rfq-cavity-lps.tn.esss.lu.se | RFQ-010:SC-IOC-017 | 172.16.100.13:47914 | DEV_RFQ-010:EMR-TT-026_iDB | IO_Error |
| RFQ-010:EMR-TT-015:HI | | | | | vm-rfq-cavity-lps.tn.esss.lu.se | RFQ-010:SC-IOC-017 | 172.16.100.13:47914 | DEV_RFQ-010:EMR-TT-015_iDB | HI |
| RFQ-010:PBI-BCM-201:Flat | | | | | pbi-bcm01-mtca-cpu-201.tn.esss.lu.se | PBI-BCM01:SC-IOC-201 | 172.16.62.177:58358 | | |
| RFQ-010:EMR-XRS-001:Hig | | | | | rfq-ipc-01.tn.esss.lu.se | RFQ-010:SC-IOC-977 | 172.16.60.179:35294 | | |
| RFQ-010:RFS-Circ-120:Wtr | tins | | | | rfq-rf-vm-ioc.tn.esss.lu.se | RFQ-010:SC-IOC-106 | 172.16.60.18:35152 | | |
| RFQ-010:RFS-CPU-101:#Se | | | | | rfq-rf-vm-ioc.tn.esss.lu.se | RFQ-010:SC-IOC-107 | 172.16.60.18:59752 | | |
| RFQ-010:RFS-DIG-101:Sen6ThrlJnCrt | anderslindh | | | | rfq-rf-vm-ioc.tn.esss.lu.se | RFQ-010:SC-IOC-107 | 172.16.60.18:59752 | | |

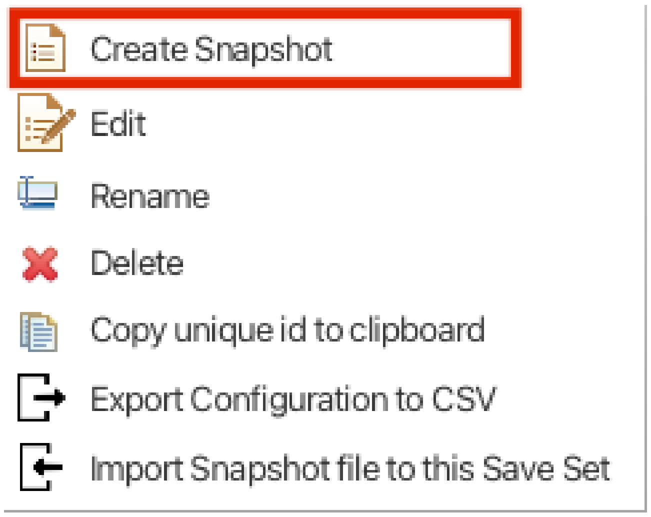
Note however that creating or updating a configuration based on a selection from the Channel Table may only populate a list of PVs. If read-back PVs are needed, they need to be added manually in the launched import dialog.

To save a configuration user must specify a (case sensitive) name and a description. Configuration names within a folder node must be unique.

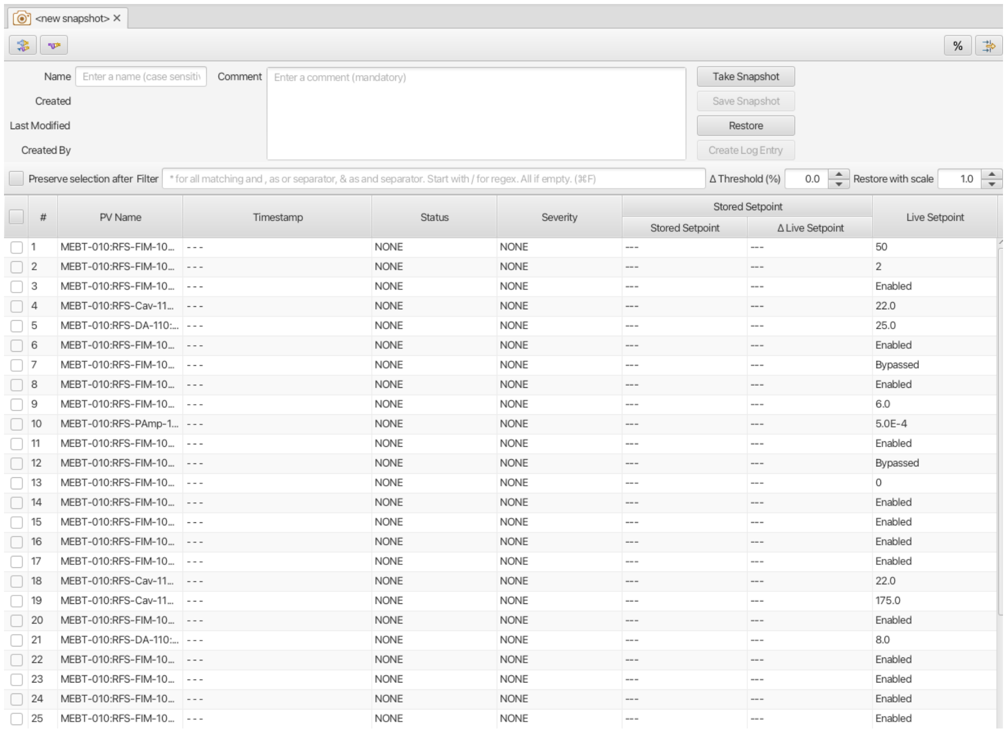
Configurations may be updated with respect to name and description. Updating the list of PVs is also supported, but user should keep in mind that existing snapshots associated with that configuration are *not* updated, e.g. PVs removed from a configurations will remain in existing snapshots.

## 8.8.10 Create Snapshot

To create a new snapshot one selects the Create Snapshot option from the context menu of a configuration:


Create Snapshot
Edit
Rename
Delete
Copy unique id to clipboard
Export Configuration to CSV
Import Snapshot file to this Save Set

This will open the snapshot view:

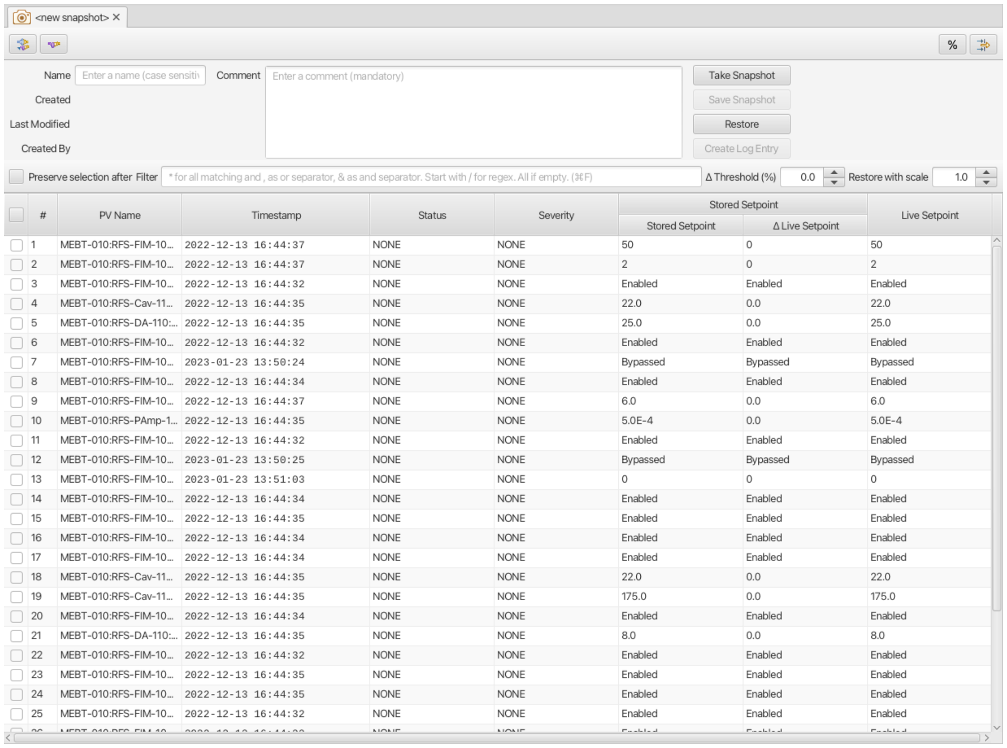


| | |
|---|---|
| Name | Enter a name (case sensiti |
| Comment | Enter a comment (mandatory) |
| Created | |
| Last Modified | |
| Created By | |
| Preserve selection after Filter | * for all matching and , as or separator, & as and separator. Start with / for regex. All if empty. (⌘F) |
| Δ Threshold (%) | 0.0 |
| Restore with scale | 1.0 |

| # | PV Name | Timestamp | Status | Severity | Stored Setpoint | | Live Setpoint |
|---|---|---|---|---|---|---|---|
| | | | | | Stored Setpoint | Δ Live Setpoint | |
| 1 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | 50 |
| 2 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | 2 |
| 3 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | Enabled |
| 4 | MEBT-010:RFS-Cav-11... | - - - | NONE | NONE | --- | --- | 22.0 |
| 5 | MEBT-010:RFS-DA-110:... | - - - | NONE | NONE | --- | --- | 25.0 |
| 6 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | Enabled |
| 7 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | Bypassed |
| 8 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | Enabled |
| 9 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | 6.0 |
| 10 | MEBT-010:RFS-PAmp-1... | - - - | NONE | NONE | --- | --- | 5.0E-4 |
| 11 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | Enabled |
| 12 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | Bypassed |
| 13 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | 0 |
| 14 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | Enabled |
| 15 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | Enabled |
| 16 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | Enabled |
| 17 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | Enabled |
| 18 | MEBT-010:RFS-Cav-11... | - - - | NONE | NONE | --- | --- | 22.0 |
| 19 | MEBT-010:RFS-Cav-11... | - - - | NONE | NONE | --- | --- | 175.0 |
| 20 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | Enabled |
| 21 | MEBT-010:RFS-DA-110:... | - - - | NONE | NONE | --- | --- | 8.0 |
| 22 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | Enabled |
| 23 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | Enabled |
| 24 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | Enabled |
| 25 | MEBT-010:RFS-FIM-10... | - - - | NONE | NONE | --- | --- | Enabled |

The left-most column will show live values for the list of PVs in the configuration. If the application fails to connect to a PV, this will be indicated accordingly.

Clicking the Take Snapshot button will disable the UI while all PVs are read. Once the read operation completes, values are displayed in the view:



Name: Enter a name (case sensiti
Comment: Enter a comment (mandatory)
Created
Last Modified
Created By

Preserve selection after Filter: * for all matching and , as or separator, & as and separator. Start with / for regex. All if empty. (⌘F)
Δ Threshold (%): 0.0
Restore with scale: 1.0

| | # | PV Name | Timestamp | Status | Severity | Stored Setpoint | | Live Setpoint |
|---|---|---|---|---|---|---|---|---|
| | | | | | | Stored Setpoint | Δ Live Setpoint | |
| ☐ | 1 | MEBT-010:RFS-FIM-10... | 2022-12-13 16:44:37 | NONE | NONE | 50 | 0 | 50 |
| ☐ | 2 | MEBT-010:RFS-FIM-10... | 2022-12-13 16:44:37 | NONE | NONE | 2 | 0 | 2 |
| ☐ | 3 | MEBT-010:RFS-FIM-10... | 2022-12-13 16:44:32 | NONE | NONE | Enabled | Enabled | Enabled |
| ☐ | 4 | MEBT-010:RFS-Cav-11... | 2022-12-13 16:44:35 | NONE | NONE | 22.0 | 0.0 | 22.0 |
| ☐ | 5 | MEBT-010:RFS-DA-110:... | 2022-12-13 16:44:35 | NONE | NONE | 25.0 | 0.0 | 25.0 |
| ☐ | 6 | MEBT-010:RFS-FIM-10... | 2022-12-13 16:44:32 | NONE | NONE | Enabled | Enabled | Enabled |
| ☐ | 7 | MEBT-010:RFS-FIM-10... | 2023-01-23 13:50:24 | NONE | NONE | Bypassed | Bypassed | Bypassed |
| ☐ | 8 | MEBT-010:RFS-FIM-10... | 2022-12-13 16:44:34 | NONE | NONE | Enabled | Enabled | Enabled |
| ☐ | 9 | MEBT-010:RFS-FIM-10... | 2022-12-13 16:44:37 | NONE | NONE | 6.0 | 0.0 | 6.0 |
| ☐ | 10 | MEBT-010:RFS-PAmp-1... | 2022-12-13 16:44:35 | NONE | NONE | 5.0E-4 | 0.0 | 5.0E-4 |
| ☐ | 11 | MEBT-010:RFS-FIM-10... | 2022-12-13 16:44:32 | NONE | NONE | Enabled | Enabled | Enabled |
| ☐ | 12 | MEBT-010:RFS-FIM-10... | 2023-01-23 13:50:25 | NONE | NONE | Bypassed | Bypassed | Bypassed |
| ☐ | 13 | MEBT-010:RFS-FIM-10... | 2023-01-23 13:51:03 | NONE | NONE | 0 | 0 | 0 |
| ☐ | 14 | MEBT-010:RFS-FIM-10... | 2022-12-13 16:44:34 | NONE | NONE | Enabled | Enabled | Enabled |
| ☐ | 15 | MEBT-010:RFS-FIM-10... | 2022-12-13 16:44:35 | NONE | NONE | Enabled | Enabled | Enabled |
| ☐ | 16 | MEBT-010:RFS-FIM-10... | 2022-12-13 16:44:34 | NONE | NONE | Enabled | Enabled | Enabled |
| ☐ | 17 | MEBT-010:RFS-FIM-10... | 2022-12-13 16:44:34 | NONE | NONE | Enabled | Enabled | Enabled |
| ☐ | 18 | MEBT-010:RFS-Cav-11... | 2022-12-13 16:44:35 | NONE | NONE | 22.0 | 0.0 | 22.0 |
| ☐ | 19 | MEBT-010:RFS-Cav-11... | 2022-12-13 16:44:35 | NONE | NONE | 175.0 | 0.0 | 175.0 |
| ☐ | 20 | MEBT-010:RFS-FIM-10... | 2022-12-13 16:44:34 | NONE | NONE | Enabled | Enabled | Enabled |
| ☐ | 21 | MEBT-010:RFS-DA-110:... | 2022-12-13 16:44:35 | NONE | NONE | 8.0 | 0.0 | 8.0 |
| ☐ | 22 | MEBT-010:RFS-FIM-10... | 2022-12-13 16:44:32 | NONE | NONE | Enabled | Enabled | Enabled |
| ☐ | 23 | MEBT-010:RFS-FIM-10... | 2022-12-13 16:44:35 | NONE | NONE | Enabled | Enabled | Enabled |
| ☐ | 24 | MEBT-010:RFS-FIM-10... | 2022-12-13 16:44:35 | NONE | NONE | Enabled | Enabled | Enabled |
| ☐ | 25 | MEBT-010:RFS-FIM-10... | 2022-12-13 16:44:32 | NONE | NONE | Enabled | Enabled | Enabled |

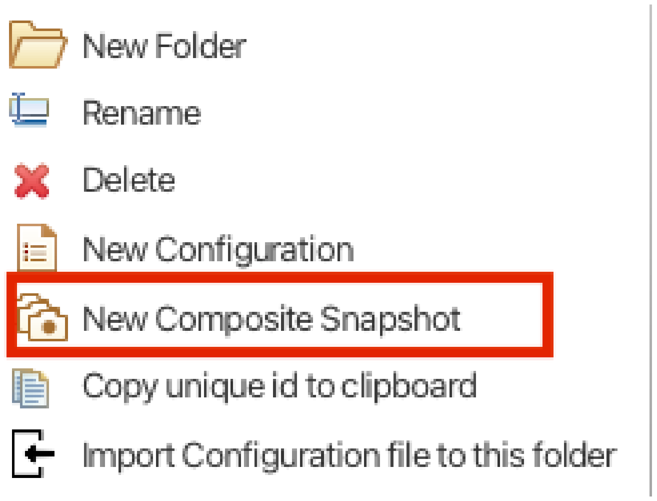
Note that the Timestamp column shows the timestamp as provided by the PV record, i.e. it need not be the current timestamp.

Once a snapshot has been taken, user must provide a case sensitive name and comment to be able to save it. Snapshot names for the same configuration must be unique. User may choose to take a new snapshot in the same view before saving it. Note that for a configuration with a large number of PVs the save operation may take some time, during which the UI is disabled.

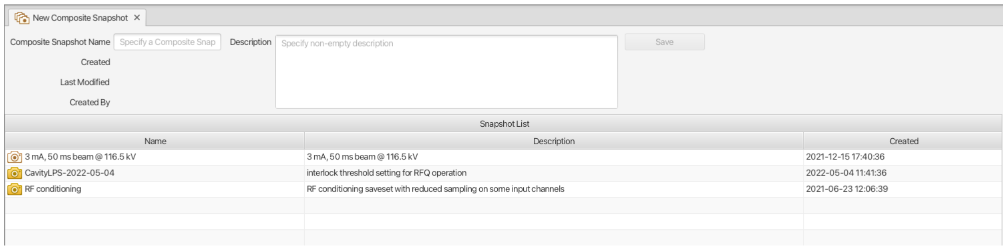
### 8.8.11 Create Composite Snapshot

A composite snapshot is an aggregation of existing snapshots or other composite snapshots, or both. Composite snapshots are **not** associated with a configuration. Instead the “configuration” - i.e. list of PVs - is implied by the list of referenced snapshots.

To create a composite snapshot user must select the New Composite Snapshot context menu option of a folder node into which the composite snapshot will be saved:


New Folder
Rename
Delete
New Configuration
New Composite Snapshot
Copy unique id to clipboard
Import Configuration file to this folder

This launches the composite snapshot editor:



| Name | Description | Created |
|---|---|---|
| 3 mA, 50 ms beam @ 116.5 kV | 3 mA, 50 ms beam @ 116.5 kV | 2021-12-15 17:40:36 |
| CavityLPS-2022-05-04 | interlock threshold setting for RFQ operation | 2022-05-04 11:41:36 |
| RF conditioning | RF conditioning saveset with reduced sampling on some input channels | 2021-06-23 12:06:39 |

Snapshot or composite snapshot items can be added to the list view in the editor by dragging wanted objects from the tree view and dropping them in the list.

The composite snapshot can be saved when a case sensitive name and a description has been specified.

**NOTE:** There are a few business rules to consider when managing composite snapshots:

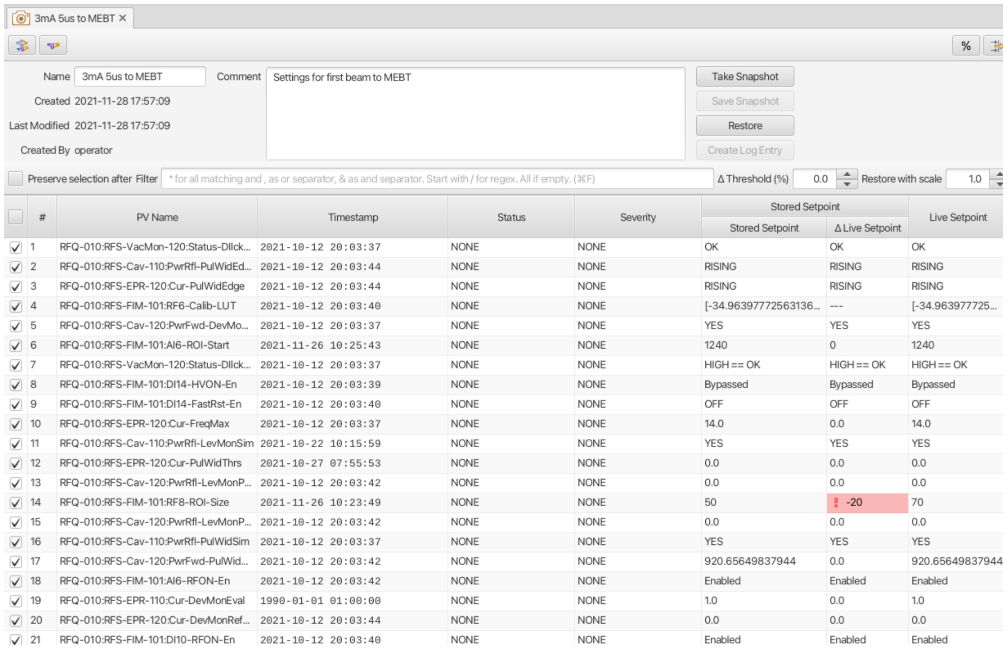
- The combined list of PV names in the referenced snapshots must not contain duplicates. This is checked for each item dropped into the list when editing a composite snapshot. If duplicates are detected, an error dialog is shown.
- Snapshots and composite snapshots cannot be deleted if referenced in a composite snapshot.

#### Edit Composite Snapshot using drag-n-drop

From the Search And Filter view (see below) user may select snapshots or composite snapshots and then drag-n-drop the selection onto an existing composite snapshot in the left-hand side tree view.

### 8.8.12 Restore Snapshot View

To open a snapshot to perform a restore operation, one must double-click on a snapshot node in the tree view. This will open the snapshot in "restore" mode, i.e. the Restore button is enabled:

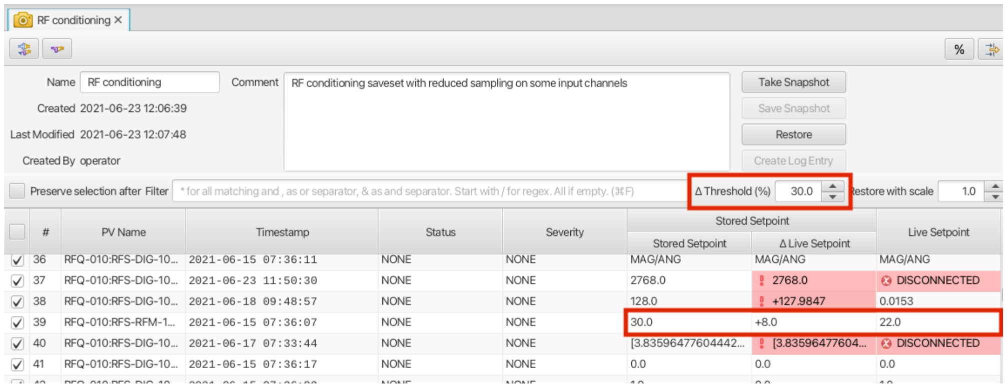


Name: 3mA 5us to MEBT
Comment: Settings for first beam to MEBT
Created: 2021-11-28 17:57:09
Last Modified: 2021-11-28 17:57:09
Created By: operator
Δ Threshold (%): 0.0
Restore with scale: 1.0

| | # | PV Name | Timestamp | Status | Severity | Stored Setpoint | | Live Setpoint |
|---|---|---|---|---|---|---|---|---|
| | | | | | | Stored Setpoint | Δ Live Setpoint | |
| ✓ | 1 | RFQ-010:RFS-VacMon-120:Status-DIlck... | 2021-10-12 20:03:37 | NONE | NONE | OK | OK | OK |
| ✓ | 2 | RFQ-010:RFS-Cav-110:PwrRfl-PulWidEd... | 2021-10-12 20:03:44 | NONE | NONE | RISING | RISING | RISING |
| ✓ | 3 | RFQ-010:RFS-EPR-120:Cur-PulWidEdge | 2021-10-12 20:03:44 | NONE | NONE | RISING | RISING | RISING |
| ✓ | 4 | RFQ-010:RFS-FIM-101:RF6-Calib-LUT | 2021-10-12 20:03:40 | NONE | NONE | [-34.96397772563136... | --- | [-34.963977725... |
| ✓ | 5 | RFQ-010:RFS-Cav-120:PwrFwd-DevMo... | 2021-10-12 20:03:37 | NONE | NONE | YES | YES | YES |
| ✓ | 6 | RFQ-010:RFS-FIM-101:AI6-ROI-Start | 2021-11-26 10:25:43 | NONE | NONE | 1240 | 0 | 1240 |
| ✓ | 7 | RFQ-010:RFS-VacMon-120:Status-DIlck... | 2021-10-12 20:03:37 | NONE | NONE | HIGH == OK | HIGH == OK | HIGH == OK |
| ✓ | 8 | RFQ-010:RFS-FIM-101:DI14-HVON-En | 2021-10-12 20:03:39 | NONE | NONE | Bypassed | Bypassed | Bypassed |
| ✓ | 9 | RFQ-010:RFS-FIM-101:DI14-FastRst-En | 2021-10-12 20:03:40 | NONE | NONE | OFF | OFF | OFF |
| ✓ | 10 | RFQ-010:RFS-EPR-120:Cur-FreqMax | 2021-10-12 20:03:37 | NONE | NONE | 14.0 | 0.0 | 14.0 |
| ✓ | 11 | RFQ-010:RFS-Cav-110:PwrRfl-LevMonSim | 2021-10-22 10:15:59 | NONE | NONE | YES | YES | YES |
| ✓ | 12 | RFQ-010:RFS-EPR-120:Cur-PulWidThrs | 2021-10-27 07:55:53 | NONE | NONE | 0.0 | 0.0 | 0.0 |
| ✓ | 13 | RFQ-010:RFS-Cav-120:PwrRfl-LevMonP... | 2021-10-12 20:03:42 | NONE | NONE | 0.0 | 0.0 | 0.0 |
| ✓ | 14 | RFQ-010:RFS-FIM-101:RF8-ROI-Size | 2021-11-26 10:23:49 | NONE | NONE | 50 | ! -20 | 70 |
| ✓ | 15 | RFQ-010:RFS-Cav-120:PwrRfl-LevMonP... | 2021-10-12 20:03:42 | NONE | NONE | 0.0 | 0.0 | 0.0 |
| ✓ | 16 | RFQ-010:RFS-Cav-110:PwrRfl-PulWidSim | 2021-10-12 20:03:37 | NONE | NONE | YES | YES | YES |
| ✓ | 17 | RFQ-010:RFS-Cav-120:PwrFwd-PulWid... | 2021-10-12 20:03:42 | NONE | NONE | 920.65649837944 | 0.0 | 920.65649837944 |
| ✓ | 18 | RFQ-010:RFS-FIM-101:AI6-RFON-En | 2021-10-12 20:03:42 | NONE | NONE | Enabled | Enabled | Enabled |
| ✓ | 19 | RFQ-010:RFS-EPR-110:Cur-DevMonEval | 1990-01-01 01:00:00 | NONE | NONE | 1.0 | 0.0 | 1.0 |
| ✓ | 20 | RFQ-010:RFS-EPR-120:Cur-DevMonRef... | 2021-10-12 20:03:44 | NONE | NONE | 0.0 | 0.0 | 0.0 |
| ✓ | 21 | RFQ-010:RFS-FIM-101:DI10-RFON-En | 2021-10-12 20:03:40 | NONE | NONE | Enabled | Enabled | Enabled |

As seen from the screenshot, the Δ Live Setpoint column highlights PVs where there is a difference $\neq 0$ between stored and live values. For array PVs the comparison is made element by element. For PV types where showing a difference is difficult (e.g. arrays) or not meaningful (booleans, enums), this column shows a suitable message instead of a Δ value.

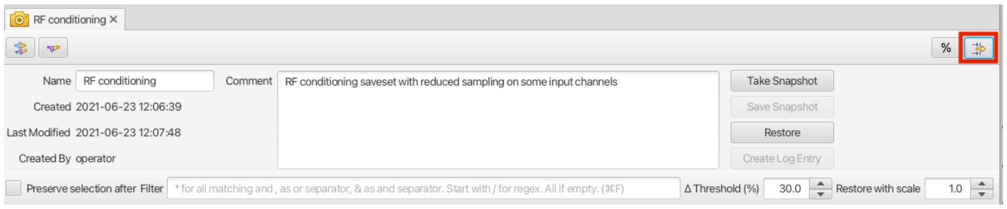
User may choose to suppress highlighting of Δ values $\neq 0$ for scalar data type PVs by specifying a threshold value:



Name: RF conditioning
Comment: RF conditioning saveset with reduced sampling on some input channels
Created: 2021-06-23 12:06:39
Last Modified: 2021-06-23 12:07:48
Created By: operator
Δ Threshold (%): 30.0
Restore with scale: 1.0

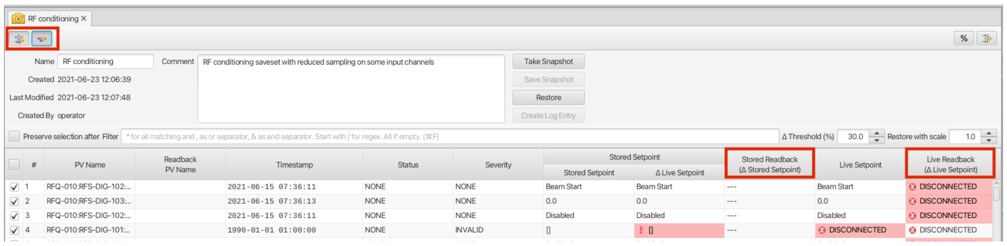
| | # | PV Name | Timestamp | Status | Severity | Stored Setpoint | | Live Setpoint |
|---|---|---|---|---|---|---|---|---|
| | | | | | | Stored Setpoint | Δ Live Setpoint | |
| ✓ | 36 | RFQ-010:RFS-DIG-10... | 2021-06-15 07:36:11 | NONE | NONE | MAG/ANG | MAG/ANG | MAG/ANG |
| ✓ | 37 | RFQ-010:RFS-DIG-10... | 2021-06-23 11:50:30 | NONE | NONE | 2768.0 | ! 2768.0 | DISCONNECTED |
| ✓ | 38 | RFQ-010:RFS-DIG-10... | 2021-06-18 09:48:57 | NONE | NONE | 128.0 | ! +127.9847 | 0.0153 |
| ✓ | 39 | RFQ-010:RFS-RFM-1... | 2021-06-15 07:36:07 | NONE | NONE | 30.0 | +8.0 | 22.0 |
| ✓ | 40 | RFQ-010:RFS-DIG-10... | 2021-06-17 07:33:44 | NONE | NONE | [3.83596477604442... | ! [3.83596477604... | DISCONNECTED |
| ✓ | 41 | RFQ-010:RFS-DIG-10... | 2021-06-15 07:36:17 | NONE | NONE | 0.0 | 0.0 | 0.0 |

It is also possible to hide all PV items where the stored value is equal to live value. The right-most button in the toolbar is used to toggle between show/hide:



Name: RF conditioning
Comment: RF conditioning saveset with reduced sampling on some input channels
Created: 2021-06-23 12:06:39
Last Modified: 2021-06-23 12:07:48
Created By: operator
Δ Threshold (%): 30.0
Restore with scale: 1.0

The snapshot view does by default not show PV read-back values if such have been defined in the configuration. The left-most columns in the toolbar can be used to show/hide columns associated with such read-back PVs:



Name RF conditioning
Comment RF conditioning saveset with reduced sampling on some input channels
Created 2021-06-23 12:06:39
Last Modified 2021-06-23 12:07:48
Created By operator
Preserve selection after Filter * for all matching and , as or separator, & as and separator. Start with / for regex. All if empty. (⌘F)
Δ Threshold (%) 30.0 Restore with scale 1.0

| | # | PV Name | Readback PV Name | Timestamp | Status | Severity | Stored Setpoint | | Stored Readback (Δ Stored Setpoint) | Live Setpoint | Live Readback (Δ Live Setpoint) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | Stored Setpoint | Δ Live Setpoint | | | |
| ✓ | 1 | RFQ-010:RFS-DIG-102:... | | 2021-06-15 07:36:11 | NONE | NONE | Beam Start | Beam Start | --- | Beam Start | DISCONNECTED |
| ✓ | 2 | RFQ-010:RFS-DIG-103:... | | 2021-06-15 07:36:13 | NONE | NONE | 0.0 | 0.0 | --- | 0.0 | DISCONNECTED |
| ✓ | 3 | RFQ-010:RFS-DIG-102:... | | 2021-06-15 07:36:11 | NONE | NONE | Disabled | Disabled | --- | Disabled | DISCONNECTED |
| ✓ | 4 | RFQ-010:RFS-DIG-101:... | | 1990-01-01 01:00:00 | NONE | INVALID | [] | ! [] | --- | DISCONNECTED | DISCONNECTED |

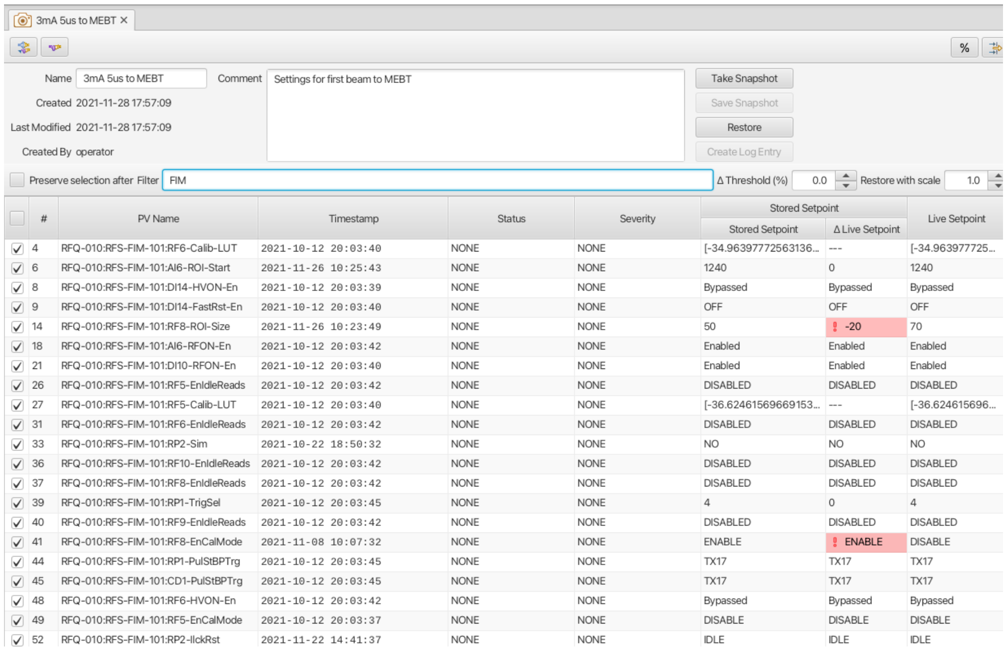
## 8.8.13 Restoring A Snapshot

To restore the values in the snapshot user should click the Restore button. During the restore operation the UI is disabled. In case a write operation fails on a PV, the process is *not* aborted, and user will be presented with a list of failed PVs when restore operation completes.

**NOTE:** During the restore operation PVs are written in parallel, i.e. in an asynchronous manner. The order of write operations is hence undefined.

Prior to restore user has the option to:

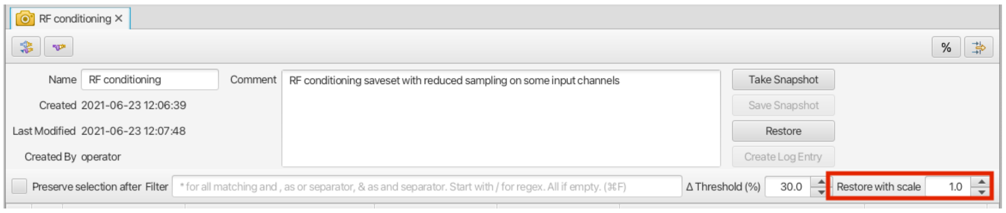
- Exclude PVs using the checkboxes in the left-most column. To simplify selection, user may use the Filter input field to find matching PV names:



Name 3mA 5us to MEBT
Comment Settings for first beam to MEBT
Created 2021-11-28 17:57:09
Last Modified 2021-11-28 17:57:09
Created By operator
Preserve selection after Filter FIM
Δ Threshold (%) 0.0 Restore with scale 1.0

| | # | PV Name | Timestamp | Status | Severity | Stored Setpoint | | Live Setpoint |
|---|---|---|---|---|---|---|---|---|
| | | | | | | Stored Setpoint | Δ Live Setpoint | |
| ✓ | 4 | RFQ-010:RFS-FIM-101:RF6-Calib-LUT | 2021-10-12 20:03:40 | NONE | NONE | [-34.96397772563136... | --- | [-34.963977725... |
| ✓ | 6 | RFQ-010:RFS-FIM-101:AI6-ROI-Start | 2021-11-26 10:25:43 | NONE | NONE | 1240 | 0 | 1240 |
| ✓ | 8 | RFQ-010:RFS-FIM-101:DI14-HVON-En | 2021-10-12 20:03:39 | NONE | NONE | Bypassed | Bypassed | Bypassed |
| ✓ | 9 | RFQ-010:RFS-FIM-101:DI14-FastRst-En | 2021-10-12 20:03:40 | NONE | NONE | OFF | OFF | OFF |
| ✓ | 14 | RFQ-010:RFS-FIM-101:RF8-ROI-Size | 2021-11-26 10:23:49 | NONE | NONE | 50 | ! -20 | 70 |
| ✓ | 18 | RFQ-010:RFS-FIM-101:AI6-RFON-En | 2021-10-12 20:03:42 | NONE | NONE | Enabled | Enabled | Enabled |
| ✓ | 21 | RFQ-010:RFS-FIM-101:DI10-RFON-En | 2021-10-12 20:03:40 | NONE | NONE | Enabled | Enabled | Enabled |
| ✓ | 26 | RFQ-010:RFS-FIM-101:RF5-EnIdleReads | 2021-10-12 20:03:42 | NONE | NONE | DISABLED | DISABLED | DISABLED |
| ✓ | 27 | RFQ-010:RFS-FIM-101:RF5-Calib-LUT | 2021-10-12 20:03:40 | NONE | NONE | [-36.62461569669153... | --- | [-36.624615696... |
| ✓ | 31 | RFQ-010:RFS-FIM-101:RF6-EnIdleReads | 2021-10-12 20:03:42 | NONE | NONE | DISABLED | DISABLED | DISABLED |
| ✓ | 33 | RFQ-010:RFS-FIM-101:RP2-Sim | 2021-10-22 18:50:32 | NONE | NONE | NO | NO | NO |
| ✓ | 36 | RFQ-010:RFS-FIM-101:RF10-EnIdleReads | 2021-10-12 20:03:42 | NONE | NONE | DISABLED | DISABLED | DISABLED |
| ✓ | 37 | RFQ-010:RFS-FIM-101:RF8-EnIdleReads | 2021-10-12 20:03:42 | NONE | NONE | DISABLED | DISABLED | DISABLED |
| ✓ | 39 | RFQ-010:RFS-FIM-101:RP1-TrigSel | 2021-10-12 20:03:45 | NONE | NONE | 4 | 0 | 4 |
| ✓ | 40 | RFQ-010:RFS-FIM-101:RF9-EnIdleReads | 2021-10-12 20:03:42 | NONE | NONE | DISABLED | DISABLED | DISABLED |
| ✓ | 41 | RFQ-010:RFS-FIM-101:RF8-EnCalMode | 2021-11-08 10:07:32 | NONE | NONE | ENABLE | ! ENABLE | DISABLE |
| ✓ | 44 | RFQ-010:RFS-FIM-101:RP1-PulStBPTrg | 2021-10-12 20:03:45 | NONE | NONE | TX17 | TX17 | TX17 |
| ✓ | 45 | RFQ-010:RFS-FIM-101:CD1-PulStBPTrg | 2021-10-12 20:03:45 | NONE | NONE | TX17 | TX17 | TX17 |
| ✓ | 48 | RFQ-010:RFS-FIM-101:RF6-HVON-En | 2021-10-12 20:03:42 | NONE | NONE | Bypassed | Bypassed | Bypassed |
| ✓ | 49 | RFQ-010:RFS-FIM-101:RF5-EnCalMode | 2021-10-12 20:03:37 | NONE | NONE | DISABLE | DISABLE | DISABLE |
| ✓ | 52 | RFQ-010:RFS-FIM-101:RP2-IlckRst | 2021-11-22 14:41:37 | NONE | NONE | IDLE | IDLE | IDLE |

- Specify a multiplier factor $\neq 1$ applied to scalar data type PVs:

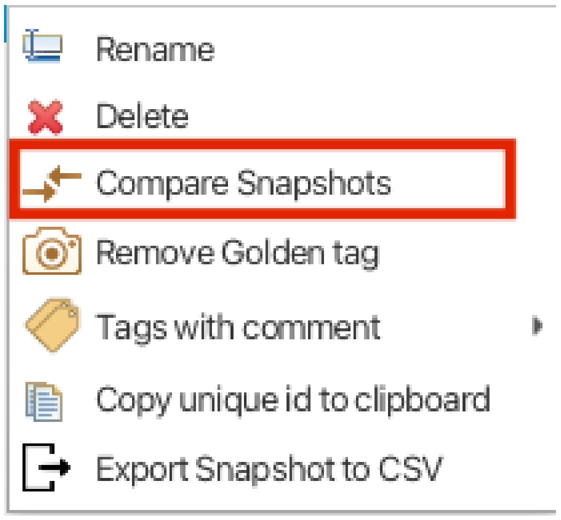


Name RF conditioning
Comment RF conditioning saveset with reduced sampling on some input channels
Created 2021-06-23 12:06:39
Last Modified 2021-06-23 12:07:48
Created By operator
Preserve selection after Filter * for all matching and , as or separator, & as and separator. Start with / for regex. All if empty. (⌘F)
Δ Threshold (%) 30.0 Restore with scale 1.0

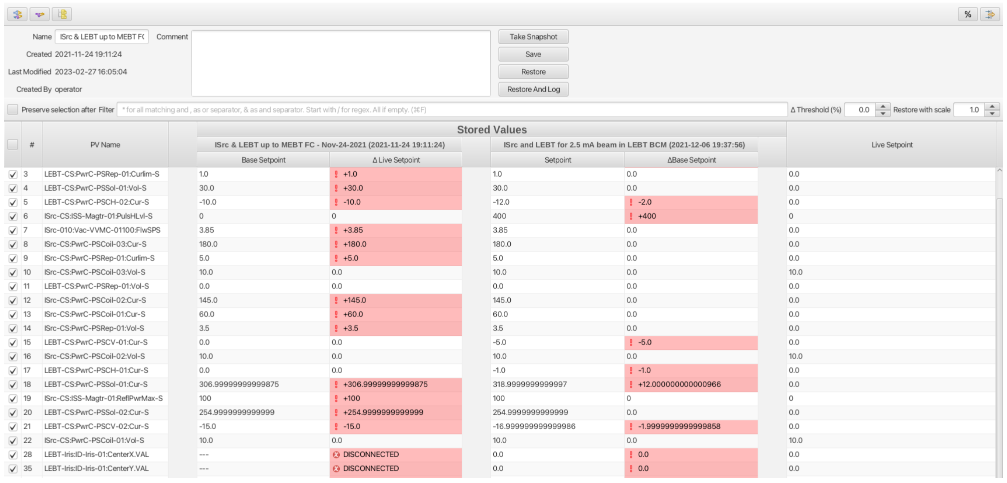
Restoring from a composite snapshot works in the same manner as the restore operation from a single-snapshot.

### 8.8.14 Comparing Snapshots

To compare two (or more) snapshots, user must first open an existing snapshot (double click in tree view). Using the Compare Snapshots context menu item for a snapshot node user may choose a snapshot to load for comparison:



Once the additional snapshot has been loaded, the snapshot view will show stored values from both snapshots. In this view the Δ Base Snapshot column will show the difference to the reference snapshot values:



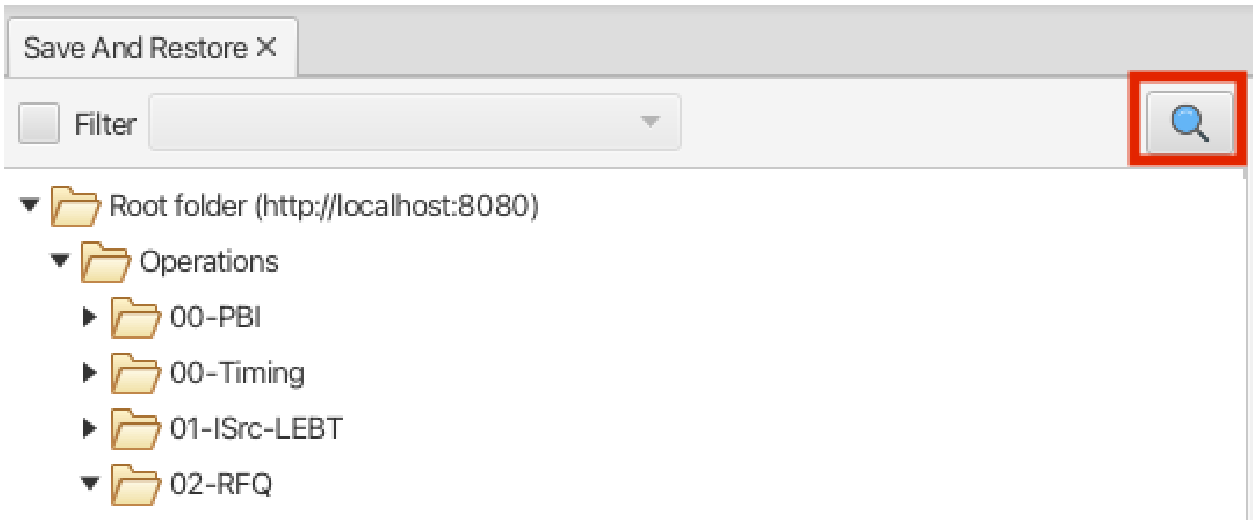
Name ISrc & LEBT up to MEBT F(
Created 2021-11-24 19:11:24
Last Modified 2023-02-27 16:05:04
Created By operator
Comment

Preserve selection after Filter * for all matching and , as or separator, & as and separator. Start with / for regex. All if empty. (⌘F)
Δ Threshold (%) 0.0 Restore with scale 1.0

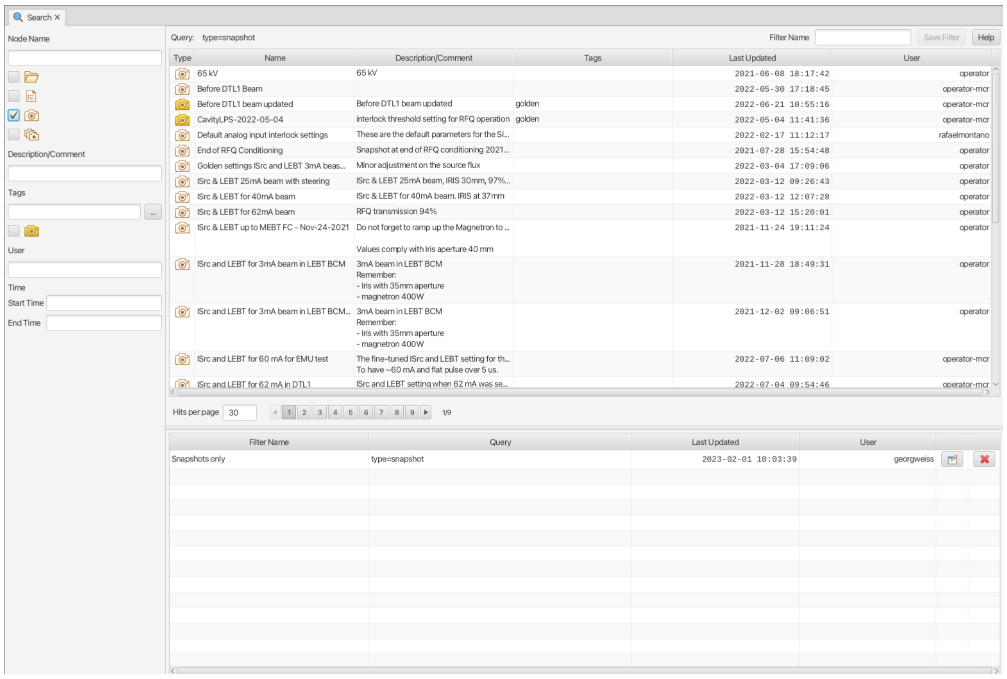
|  |  |  | Stored Values |  |  |  |  |
|---|---|---|---|---|---|---|---|
|  |  |  | ISrc & LEBT up to MEBT FC - Nov-24-2021 (2021-11-24 19:11:24) |  | ISrc and LEBT for 2.5 mA beam in LEBT BCM (2021-12-06 19:37:56) |  |  |
|  | # | PV Name | Base Setpoint | Δ Live Setpoint | Setpoint | ΔBase Setpoint | Live Setpoint |
| ✓ | 3 | LEBT-CS:PwrC-PSRep-01:Curlim-S | 1.0 | +1.0 | 1.0 | 0.0 | 0.0 |
| ✓ | 4 | LEBT-CS:PwrC-PSSol-01:Vol-S | 30.0 | +30.0 | 30.0 | 0.0 | 0.0 |
| ✓ | 5 | LEBT-CS:PwrC-PSCH-02:Cur-S | -10.0 | -10.0 | -12.0 | -2.0 | 0.0 |
| ✓ | 6 | ISrc-CS:ISS-Magtr-01:PulsHLvl-S | 0 | 0 | 400 | +400 | 0 |
| ✓ | 7 | ISrc-010:Vac-VVMC-01100:FlwSPS | 3.85 | +3.85 | 3.85 | 0.0 | 0.0 |
| ✓ | 8 | ISrc-CS:PwrC-PSCoil-03:Cur-S | 180.0 | +180.0 | 180.0 | 0.0 | 0.0 |
| ✓ | 9 | ISrc-CS:PwrC-PSRep-01:Curlim-S | 5.0 | +5.0 | 5.0 | 0.0 | 0.0 |
| ✓ | 10 | ISrc-CS:PwrC-PSCoil-03:Vol-S | 10.0 | 0.0 | 10.0 | 0.0 | 10.0 |
| ✓ | 11 | LEBT-CS:PwrC-PSRep-01:Vol-S | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| ✓ | 12 | ISrc-CS:PwrC-PSCoil-02:Cur-S | 145.0 | +145.0 | 145.0 | 0.0 | 0.0 |
| ✓ | 13 | ISrc-CS:PwrC-PSCoil-01:Cur-S | 60.0 | +60.0 | 60.0 | 0.0 | 0.0 |
| ✓ | 14 | ISrc-CS:PwrC-PSRep-01:Vol-S | 3.5 | +3.5 | 3.5 | 0.0 | 0.0 |
| ✓ | 15 | LEBT-CS:PwrC-PSCV-01:Cur-S | 0.0 | 0.0 | -5.0 | -5.0 | 0.0 |
| ✓ | 16 | ISrc-CS:PwrC-PSCoil-02:Vol-S | 10.0 | 0.0 | 10.0 | 0.0 | 10.0 |
| ✓ | 17 | LEBT-CS:PwrC-PSCH-01:Cur-S | 0.0 | 0.0 | -1.0 | -1.0 | 0.0 |
| ✓ | 18 | LEBT-CS:PwrC-PSSol-01:Cur-S | 306.99999999999875 | +306.99999999999875 | 318.9999999999997 | +12.000000000000966 | 0.0 |
| ✓ | 19 | ISrc-CS:ISS-Magtr-01:ReflPwrMax-S | 100 | +100 | 100 | 0 | 0 |
| ✓ | 20 | LEBT-CS:PwrC-PSSol-02:Cur-S | 254.9999999999999 | +254.9999999999999 | 254.9999999999999 | 0.0 | 0.0 |
| ✓ | 21 | LEBT-CS:PwrC-PSCV-02:Cur-S | -15.0 | -15.0 | -16.999999999999986 | -1.9999999999999858 | 0.0 |
| ✓ | 22 | ISrc-CS:PwrC-PSCoil-01:Vol-S | 10.0 | 0.0 | 10.0 | 0.0 | 10.0 |
| ✓ | 28 | LEBT-Iris:ID-Iris-01:CenterX.VAL | --- | DISCONNECTED | 0.0 | 0.0 | 0.0 |
| ✓ | 35 | LEBT-Iris:ID-Iris-01:CenterY.VAL | --- | DISCONNECTED | 0.0 | 0.0 | 0.0 |

### 8.8.15 Search And Filters

The search tool is launched as a separate view through the icon on top of the tree view:



The search tool is rendered as a separate tab and will always be the left-most tab in the right-hand side pane of the save&restore UI:



Query: type=snapshot

| Type | Name | Description/Comment | Tags | Last Updated | User |
|---|---|---|---|---|---|
| 📷 | 65 kV | 65 kV | | 2021-06-08 18:17:42 | operator |
| 📷 | Before DTL1 Beam | | | 2022-05-30 17:18:45 | operator-mcr |
| 📷 | Before DTL1 beam updated | Before DTL1 beam updated | golden | 2022-06-21 10:55:16 | operator-mcr |
| 📷 | CavityLPS-2022-05-04 | interlock threshold setting for RFQ operation | golden | 2022-05-04 11:41:36 | operator-mcr |
| 📷 | Default analog input interlock settings | These are the default parameters for the SI... | | 2022-02-17 11:12:17 | rafaelmontano |
| 📷 | End of RFQ Conditioning | Snapshot at end of RFQ conditioning 2021... | | 2021-07-28 15:54:48 | operator |
| 📷 | Golden settings ISrc and LEBT 3mA beas... | Minor adjustment on the source flux | | 2022-03-04 17:09:06 | operator |
| 📷 | ISrc & LEBT 25mA beam with steering | ISrc & LEBT 25mA beam, IRIS 30mm, 97%... | | 2022-03-12 09:26:43 | operator |
| 📷 | ISrc & LEBT for 40mA beam | ISrc & LEBT for 40mA beam. IRIS at 37mm | | 2022-03-12 12:07:28 | operator |
| 📷 | ISrc & LEBT for 62mA beam | RFQ transmission 94% | | 2022-03-12 15:20:01 | operator |
| 📷 | ISrc & LEBT up to MEBT FC - Nov-24-2021 | Do not forget to ramp up the Magnetron to ...<br><br>Values comply with Iris aperture 40 mm | | 2021-11-24 19:11:24 | operator |
| 📷 | ISrc and LEBT for 3mA beam in LEBT BCM | 3mA beam in LEBT BCM<br>Remember:<br>- Iris with 35mm aperture<br>- magnetron 400W | | 2021-11-28 18:49:31 | operator |
| 📷 | ISrc and LEBT for 3mA beam in LEBT BCM... | 3mA beam in LEBT BCM<br>Remember:<br>- Iris with 35mm aperture<br>- magnetron 400W | | 2021-12-02 09:06:51 | operator |
| 📷 | ISrc and LEBT for 60 mA for EMU test | The fine-tuned ISrc and LEBT setting for th...<br>To have ~60 mA and flat pulse over 5 us. | | 2022-07-06 11:09:02 | operator-mcr |
| 📷 | ISrc and LEBT for 62 mA in DTL1 | ISrc and LEBT setting when 62 mA was se... | | 2022-07-04 09:54:46 | operator-mcr |

Hits per page 30

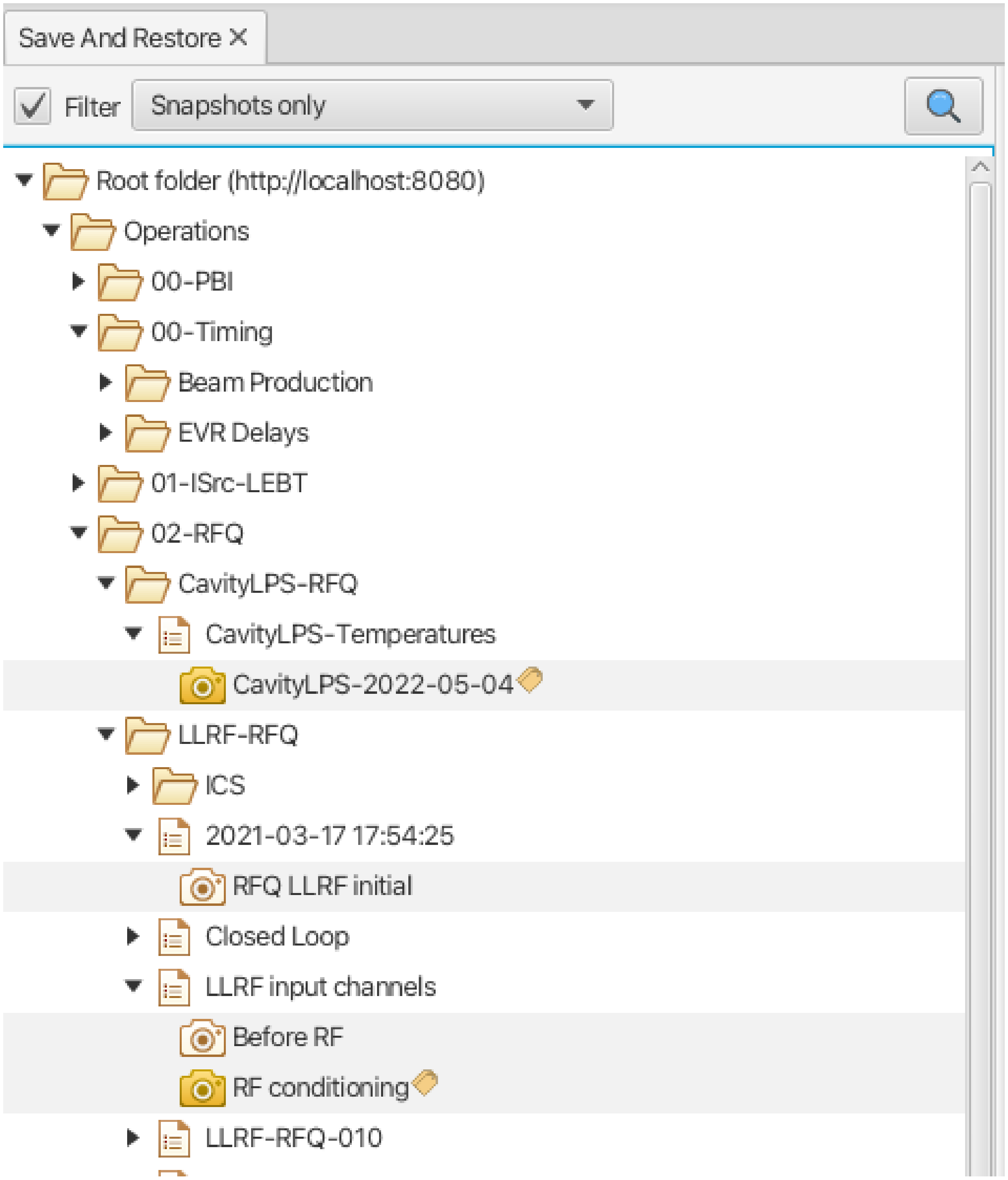
| Filter Name | Query | Last Updated | User | | |
|---|---|---|---|---|---|
| Snapshots only | type=snapshot | 2023-02-01 10:03:39 | georgweiss | | ✖ |

In the left-hand side pane user may specify criteria to match nodes. The above screen shot shows an example to search for snapshot nodes. The table on the right-hand side will show the result.

In the toolbar above the search result list user may choose to save the search query as a named "filter". The Help button will show details on how to specify the various search criteria to construct a suitable query. Filter names are case sensitive.

The bottom-right pane in the search tool shows all saved filters, which can be edited or deleted. If a filter is edited and saved it under the same name, user will be prompted whether to overwrite as filter names must be unique.

In the tree view user may select to enable and chose a saved filter:



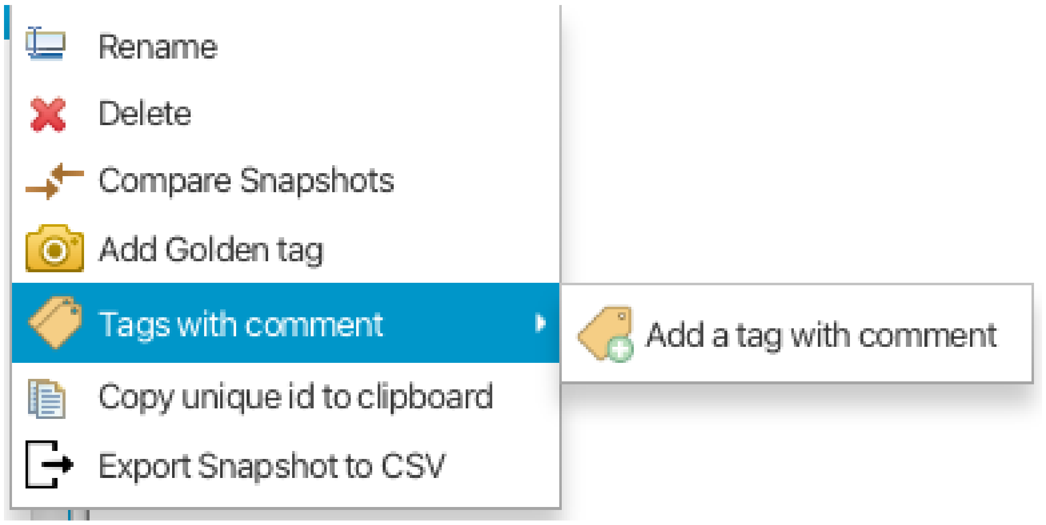
Root folder (http://localhost:8080)
Operations
00-PBI
00-Timing
Beam Production
EVR Delays
01-ISrc-LEBT
02-RFQ
CavityLPS-RFQ
CavityLPS-Temperatures
CavityLPS-2022-05-04
LLRF-RFQ
ICS
2021-03-17 17:54:25
RFQ LLRF initial
Closed Loop
LLRF input channels
Before RF
RF conditioning
LLRF-RFQ-010

Nodes in the tree view matching a filter will be highlighted, i.e. non-matching items are not hidden from the view.

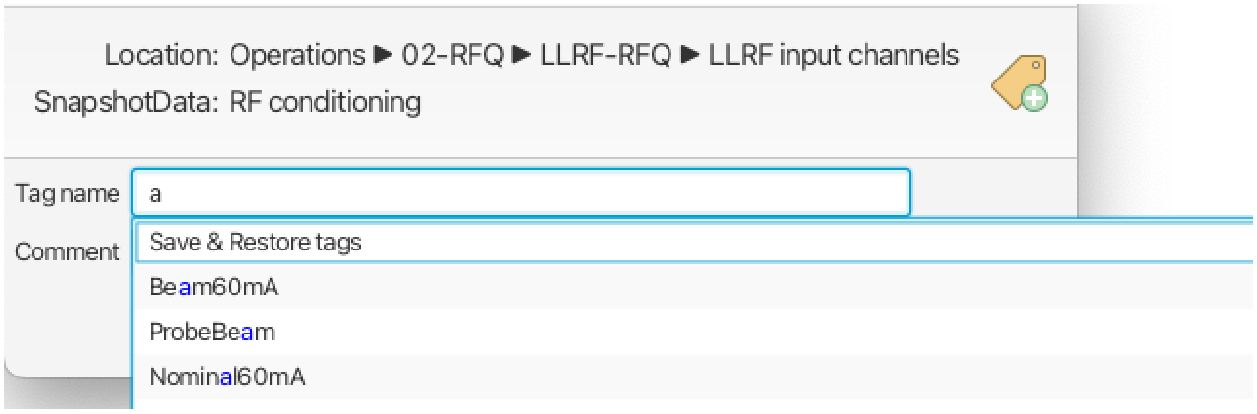
**NOTE:** When selecting a filter in the tree view, only matching items already present in the view will be highlighted. There may be additional nodes matching the current filter, but these will be rendered and highlighted only when their parent nodes are expanded. To easily find *all* matching items user will need to use the search tool.

### 8.8.16 Tagging

Tagging of snapshots can be used to facilitate search and filtering. The Tags with comment context menu option of the snapshot node is used to launch the tagging dialog:



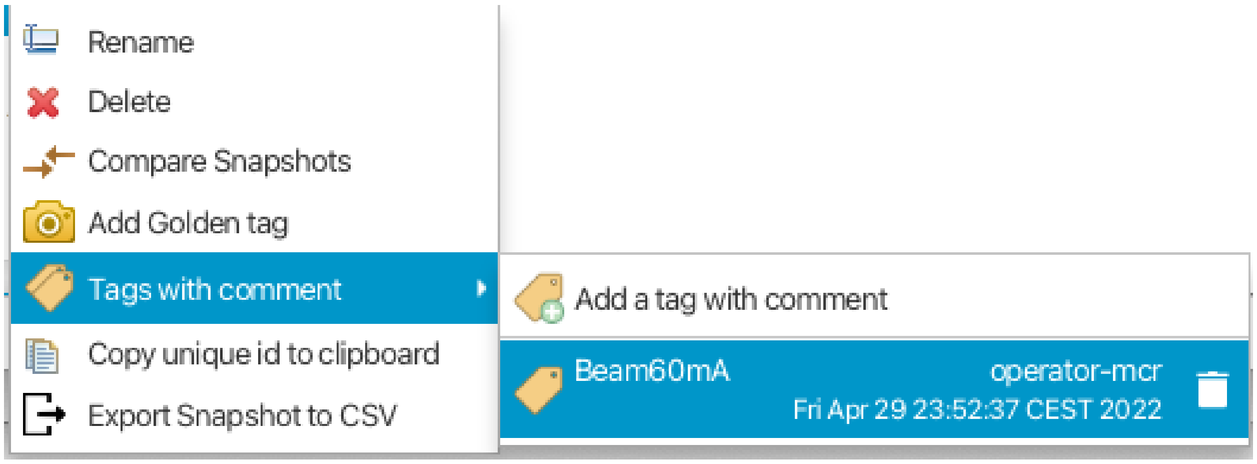
In the dialog user may specify a case sensitive tag name and optionally a comment. When typing in the Tag name field, a list of existing tag names that may match the typed text is shown. User may hence "reuse" existing tags:


Location: Operations ▶ 02-RFQ ▶ LLRF-RFQ ▶ LLRF input channels
SnapshotData: RF conditioning
Tag name a
Comment

Save & Restore tags
Beam60mA
ProbeBeam
Nominal60mA

**NOTE:** The concept of "golden" tags can be used to annotate snapshots considered to be of particular value. Such snapshots are rendered using a golden snapshot icon:

User may delete a tag through the tagging sub-menu:



### Tagging multiple snapshots

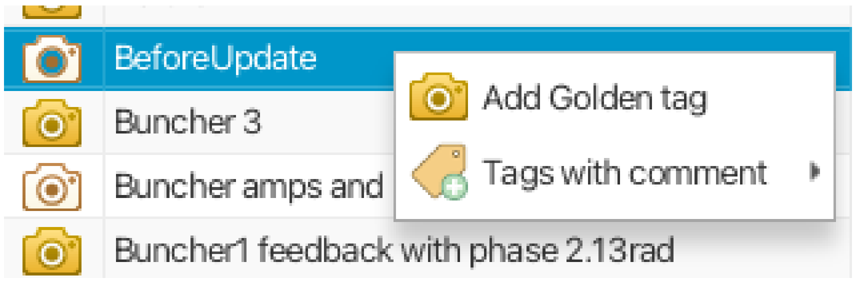
If user selects multiple snapshot nodes in the tree view, all of the selected nodes can be tagged with the same tag in one single operation. Note however that this is possible only if the wanted tag is not already present on any of the nodes.

### Deleting tags on multiple snapshots

If user selects multiple snapshot nodes, tags may be deleted on all of the nodes in one single operation. Note however that the context menu will only show tags common to all selected nodes.

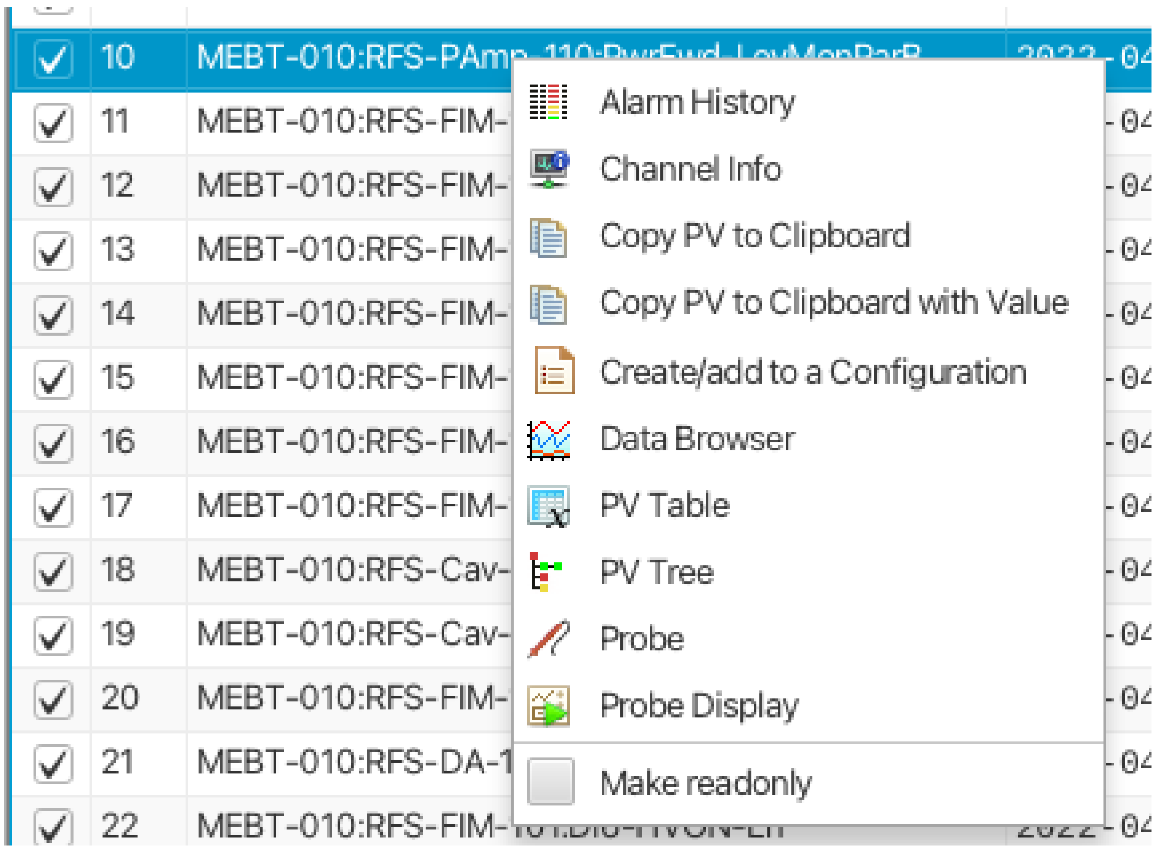
### Tagging from search view

The search result table of the Search And Filter view also supports a contect menu for the purpose of managing tags:



## 8.8.17 Snapshot View Context Menu

A right click on a table item in the restore snapshot view launches the following context menu:



The items of this context menu offers actions associated with a PV, which is similar to "PV context menus" in other applications. However, user should be aware that the "Data Browser" item will launch the Data Browser app for the selected PV *around the point in time defined by the PV timestamp*.

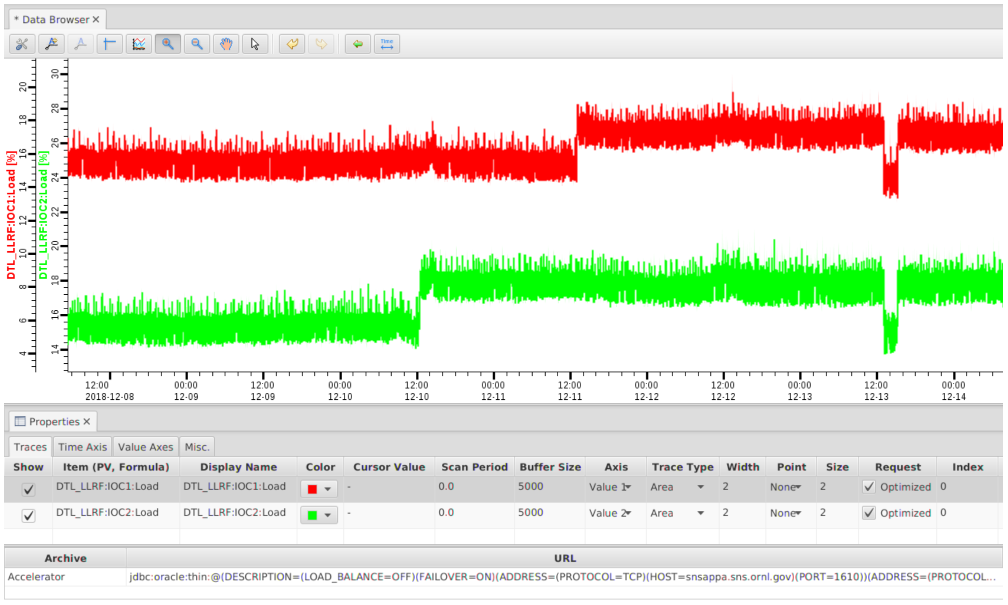
## 8.9 Data Browser

### 8.9.1 Overview

The Data Browser is a trending tool. It can display the values of 'live' Process Variables (PVs) as well as historic data in a Stripchart-type plot over time.

**Live Data** For each PV, the Data Browser collects 'live' samples from the control system and plots them over time. The size of the live sample buffer is configurable, as is the sample period.

**Historic Data** The Data Browser queries archive data sources for historic samples to obtain data for time ranges 'before' the live sample buffer. You can configure one or more archive data source for each PV.



| Show | Item (PV, Formula) | Display Name | Color | Cursor Value | Scan Period | Buffer Size | Axis | Trace Type | Width | Point | Size | Request | Index |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ✓ | DTL_LLRF:IOC1:Load | DTL_LLRF:IOC1:Load | 🟥 | - | 0.0 | 5000 | Value 1 | Area | 2 | None | 2 | ✓ Optimized | 0 |
| ✓ | DTL_LLRF:IOC2:Load | DTL_LLRF:IOC2:Load | 🟩 | - | 0.0 | 5000 | Value 2 | Area | 2 | None | 2 | ✓ Optimized | 0 |

| Archive | URL |
|---|---|
| Accelerator | jdbc:oracle:thin:@(DESCRIPTION=(LOAD_BALANCE=OFF)(FAILOVER=ON)(ADDRESS=(PROTOCOL=TCP)(HOST=snsappa.sns.ornl.gov)(PORT=1610))(ADDRESS=(PROTOCOL... |

## 8.9.2 Getting Started

To create a new plot:

1. Invoke the menu `Applications`, `Display`, `Data Browser`.
2. Open the plot's context menu by right-clicking into the plot, invoke `Add PV`.
3. Enter the desired PV name, press "OK".

## 8.9.3 Plot Configuration `.plt` Files

Use the `File`, `Save` menu entry or the keyboard shortcut CTRL + S (CMD + S on Mac OS) to save the plot settings, which include PV names, trace colors and axis arrangements. The default file extension for plot files is `.plt`. You can later re-open the configuration by opening the `.plt` file, adjust settings, for example axis ranges, and save the updated configuration.

The `.plt` files store the configuration of a plot, but no data. To save an image of the current plot, use `Save Snapshot` from the plot context menu item. To save the data shown in the plot, refer to the "Exporting Data" section below.

Note that any zooming or panning of a plot changes the plot settings, so when then closing the plot, you will be prompted to save the updated settings. This may not be desirable in an operational setup. For example, assume a `.plt` file was created with a time axis range that displays the last 6 hours. When users open this configuration, they are free to zoom in or out, even add or remove traces, but when they close the plot, you may not want them to overwrite the original settings.

There are two ways to accomplish this. One is the "Save Changes" option in the plot's "Properties Panel" described below. The other is making the `.plt` file read-only using operating system file tools or the CS-Studio File Browser. Configuration files can also be loaded via HTTP links, which are always read-only. When trying to close a plot with modified configuration, the Data Browser noticed if the original file cannot be written and will prompt with a "Save-As" dialog to offer saving to a new file.

### 8.9.4 Toolbar

Open the plot's tool bar by right-clicking into the plot, then invoke `Show Toolbar`.

#### Stagger

The `Stagger` button will set the range of each value axis such that the traces on different axes don't overlap. In other words, all the traces of the first value axis will appear on top, followed by the traces on the second axis below, the traces on the third axis below the second axis and so on.

#### Zooming

The buttons to `Zoom In` and `Zoom Out` activate the respective zoom mode.

While in `Zoom In` mode,

- pressing the mouse on a start point in the time axis, then dragging to an end point in the time axis and finally releasing the mouse button will zoom into the selected time region.
- similarly selecting a start..end range on a value axis will zoom into the selected range on that value axis.
- dragging a rectangle inside the plot will zoom into the selected region

While in `Zoom Out` mode,

- clicking the mouse on the time axis zooms out of that point in time.
- clicking the mouse on a value axis zooms out of that value on that axis.
- clicking the mouse inside the plot zooms out of that point on the time axis and all value axes.

`Zoom In` can also be activated within the plot by simply holding the `Control` key, then dragging a range on the time axis, value axis or within the plot.

Finally, while the mouse pointer is within an axis or the plot, you can hold the `Control` key and then use the scroll wheel to zoom in or out of an axis or the plot.

### 8.9.5 Properties Panel

Open the plot's property panel by right-clicking into the plot, then invoke `Open Properties Panel`.

The panel allows you to configure each trace, the time axis, the value axes, and miscellaneous settings. Changes performed in this panel will be persisted when the plot (.plt) file is saved.

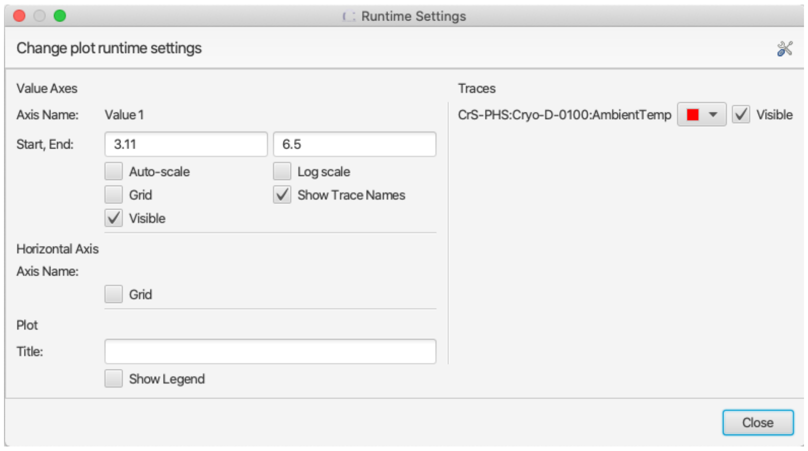
The meaning of most options should become evident by simply using them, with a few exceptions.

On the "Misc." tab, "Plot redraw period" and "Scroll Step" control how the plot updates and scrolls. For example, a redraw period of 1 and a scroll step of 10 refreshes the plot every second, while scrolling the time axis every 10 seconds. This makes for a plot that mostly stands still, only moving every 10 seconds. Larger redraw periods can be useful to reduce CPU usage.

The "Save Changes" option on the "Misc." tab controls how the plot behaves when closed. By default, the option is checked, and trying to close a modified plot will open a "Save before closing?" prompt. In an operational scenario, certain plots may be provided with default settings that are not meant to be overwritten. For those, un-check the "Save Changes" option. Users can now open the plot, zoom, pan, even add or remove traces, and then simply close the plot without a prompt, preserving the desired default settings. Note that when you un-check the "Save Changes" option to get this behavior, you already activated it, so you need to use “Save As” from the File menu to save the `*.plt` file with the un-checked “Save Changes” option.

### 8.9.6 Runtime Properties Dialog

If enabled through the preference setting `org.csstudio.trends.databrowser3/config_dialog_supported`, plot settings may also be changed using the runtime properties dialog. To launch the dialog, click on the plot area and then press letter “o”:



Runtime Settings

Change plot runtime settings

Value Axes

Axis Name: Value 1

Start, End: 3.11 6.5

☐ Auto-scale ☐ Log scale

☐ Grid ✓ Show Trace Names

✓ Visible

Horizontal Axis

Axis Name:

☐ Grid

Plot

Title:

☐ Show Legend

Traces

CrS-PHS:Cryo-D-0100:AmbientTemp 🟥 ✓ Visible

Close

**NOTE**: changes to the plot performed from the runtime properties dialog will not update the data model constructed from the underlying plot file. Consequently such changes are **not** persisted when the plot file is saved. To make sure plot properties are saved one must perform the changes in the property panel.

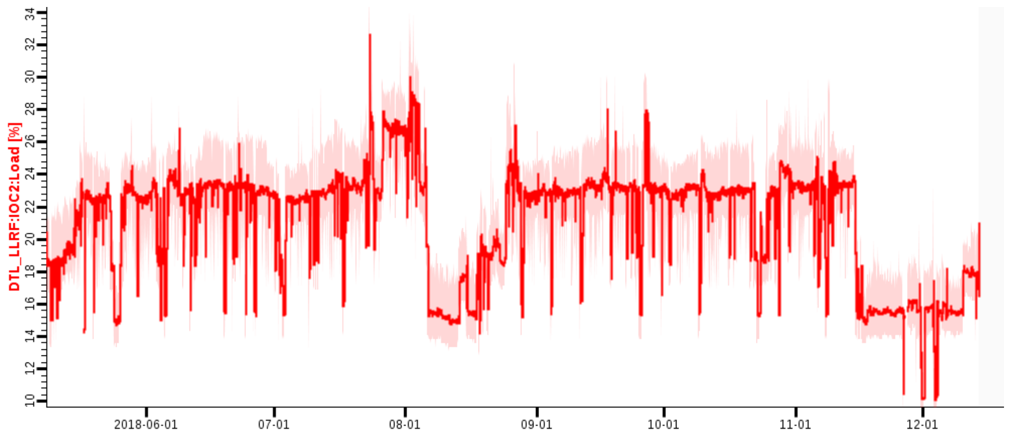
#### Live Data Sampling

By default, traces will use a `Scan Period` of 0 seconds, which means we keep every sample received from the control system. To avoid running out of memory, the live data buffer has a limited `Buffer Size`. For example, if your channel updates every 1 second, and the buffer size is set to 5000, the data browser will keep roughly the last 1 hour and 20 minutes of live data in memory. Older samples are dropped.

To keep a longer time span of live data in memory, you can increase the `Buffer Size`, which obviously results in using more memory. Alternatively, you can set the `Scan Period` to for example 5 seconds. This will ignore the update rate of the channel and simply take a sample every 5 seconds, allowing a buffer of 5000 samples to hold almost 7 hours of data.

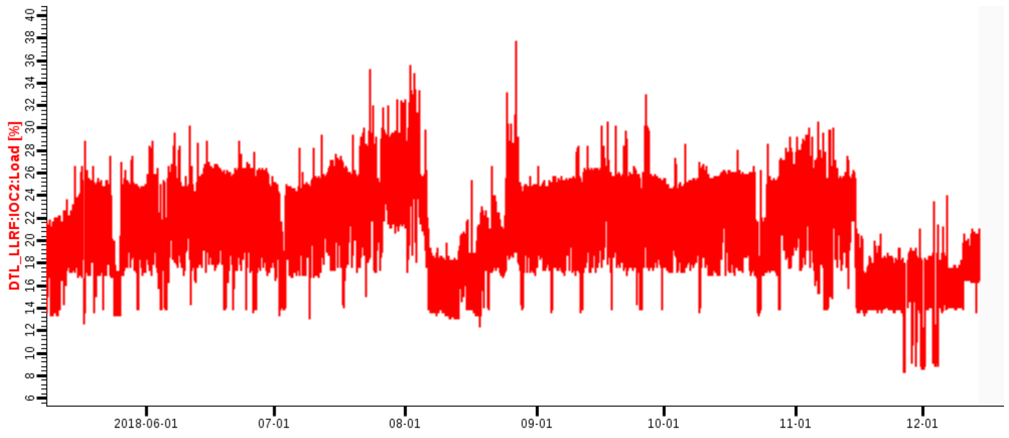
#### Optimized vs. Raw Archive Data Request

By default, a trace will request `Optimized` data from an archive. How this is accomplished depends on the implementation of the underlying archive system. The fundamental idea is that the data browser requests a reduced (optimized) set of min/max/average samples. The number of optimized samples is based on the width of the screen in pixels. When using ‘Area’ traces as shown below, the average values are plotted as a line, and the min/max outline is displayed as a light-colored area. This not only reduces the amount of data compared to plotting every single raw sample from the archive. In addition it makes it obvious if a value was fairly stable over time, or if there was a lot of movement in the data around the average. Some archive data sources may also provide the standard deviation, which will be represented as a pair of thinner lines, one above and one below the thicker average line.



When there are fewer 'raw' data points than requested, the 'Optimized' algorithm falls back to returning the raw data, meaning: When you zoom into the data far enough, you will eventually get raw data.

You may use the `Request` option ofproperty a trace in the property panel to always request `Raw` data. As a result, each original sample is fetched from the archive. The screenshot below shows the result. Not only is this usually a larger amount of samples, taking longer to receive and then to plot. It also uses more memory, and when you try to look at raw data for a long time range your computer will eventually ran out of memory. The 'Raw Data' requests should therefore only be used when necessary because of shortcomings in the 'Optimized' algorithm.

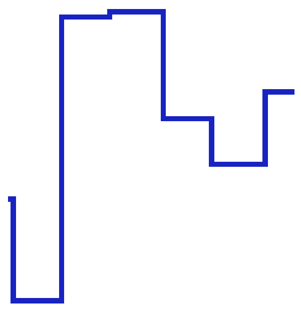


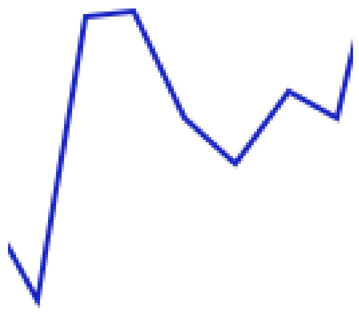
## Trace Types

Ideally, control system data is only updated when it changes by a significant amount. EPICS records actually offer separate update thresholds for live and archived data, so that live displays can receive updates for small changes at the noise level of a signal, while historic data is only written for changes well above the noise level.

When the Data Browser receives either live of historic samples, we assume that those represent significant changes in the value.

The "Area" and "Single Line" Trace Types, shown to the left in the following image, use a stair-step type of line that holds the value of the last sample until a new sample arrives. This reflects our assumption that the signal has not significantly changed until we receive a new sample.





The alternate "Area (direct)" and "Single Line (direct)" Trace Types, shown to the right, draw a direct line between samples, suggesting a linear change of the process variable between known samples.

## 8.9.7 Exporting Data

Use the "Export" panel to write data into files suitable for spreadsheet programs or Matlab. Open the export panel by right-clicking into the plot, then invoke `Open Data Export Panel`.

**Time Range** By default, the export will use the time range from the plot, but you can export data for a different time range.

**Source Options** By default, the export will fetch raw data from the archive, not the optimized, reduced min/max/average data that is displayed in the plot.

You can modify the time range of the export to differ from the range of the plot, and also select to export either the exact data that is displayed in the plot, or request optimized data with a different number of "bins" from the archive.

`Plot`: The exact samples currently displayed in the plot will be exported.

`Raw Archived Data`: The exported data will contain data that is requested from the archive without modifications.

`Optimized Archived Data`: To reduce the number of exported samples, optimized data is requested from the archive. You specify the number of desired samples, and the export will then request roughly this number of averaged samples from the archive data source. For example, when requesting 800 optimized samples, the start..end time range is split into 800 "bins". The the minimum, maximum and average value of the raw samples within each bin is then exported.

`Linear Interpolation`: As an alternative way to reduce the number of exported samples, the exported samples are computed via linear interpolation from the raw data. You specify the interpolation interval in hours, minutes and seconds, for example 00:10:00 to obtain one sample every 10 minutes.

**Format Options** The format of the exported file defaults to a spreadsheet suitable for import into most spreadsheet programs. You can modify these settings:

`Tabular`: Instead of a table that lists the samples for all channels, you can export the samples for each channel separately.

`.. with error columns`: The min…max range of optimized data is by default exported as "error" columns, suitable for an error-bar display.

`.. with Severity/Status`: The Severity and Status of each sample is by default exported, but you can omit it if not needed.

`Default format`: The number of decimal digits will be obtained from the data source.

`Decimal format`: You specify the number of decimal digits.

`Exponential notation`: Use exponential notation, where you again specify the number of decimal digits.

### Excel Spreadsheets

The exported data files are in a text format with TAB-delimited columns suitable for import into spreadsheet programs like Microsoft Excel.

Assume you chose a filename of "test.dat" on your Desktop, follow these steps for import into Excel:

- In Excel, use File/Open to open the file "test.dat". You might have to select `Files of type: All Files (*.*)` in the file "Open" dialog to do this.
- A "Text Import Wizard" should appear, and the default settings should already be set to
  - "Delimited - Characters such as … tabs …"
  - "Delimiters: Tab"
- Even though the time stamp column contains the full date and time, Excel will only display time down to minutes, omitting the seconds or microseconds. Fix this by clicking on the "A" table header, i.e. selecting the whole first column; right-click to get the "Format Cells…" dialog, and enter a "Custom" format: `yyyy/m/d h:mm:ss.000`

For plots, the "X/Y scatter" plot type using time as the X axis and the value column for the Y axis tends to work best.

Note that Excel is limited to about 65000 lines. If your data file includes more lines, those will be lost in the import. You can work around this by exporting a smaller time range into separate files, or by exporting averaged data.

The value columns for averaged data will contain text like "5 [0… 10]" to indicate that the average value for that time was 5, with a minimum/maximum range of 0 to 10. To perform computations in excel, it might be useful to select the column and perform a text replacement of "[*]" with "" (nothing) to delete the min/max info.

When performing computations on the data, values marked "#N/A" which have non numerical value because they represent a status or error should be ignored by Excel.

### OpenOffice.org/LibreOffice Calc Spreadsheets

As long as the exported data file has a ".csv" suffix, OpenOffice.org/LibreOffice should open the file with an "import" dialog similar to MS Excel, where you select "tab" delimited columns as explained above.

File names ending in ".dat", ".txt" etc. might only open in the OpenOffice.org/LibreOffice Writer (word processor), so you have to use the correct file suffix.

### Matlab

The Matlab export can use two different file formats. To create binary Matlab data files, use a file name ending in `.mat`. Matlab text files are created when the file name ends in `.m`.

The binary data file like "example.mat" is suitable for loading into Matlab 5 and newer like this:

```
load /path/to/the/example.mat
```

In your Matlab workspace you should then find structures named "channel0", "channel1", . . . , one for each channel that was loaded from the file. Each structure has the following elements:

`name`: Channel name

`time`: Time stamps

`value`: Numeric value

`severity`: Severity

The time stamps are saved as text by default. In reasonable modern versions of Matlab you best convert them to Matlab serial date numbers, which you can then for example plot over time like this:

```
channel0.date = datenum(channel0.time, 'yyyy-mm-dd HH:MM:SS.FFF');
plot(channel0.date, channel0.value);
datetick('x',0);
title(channel0.name);
```

If user ticks 'Use UNIX timestamp. . . ' in the export tab, timestamps will be exported as UNIX timestamp in ms, i.e. as a 64 bit unsigned integer.

The Matlab text files like "example.m" are suitable for loading (executing) in Matlab R2006b or newer, creating one 'timeseries' object per trace. To load it into Matlab, you execute the file like this:

```
cd /path/to/the/file
example
```

Matlab will then execute the commands in the file which define the data in the Matlab workspace and also plot it. With Matlab releases older than R2006b you will have to edit the generated file to suit your needs, for example use simply `plot(v)` to show the values.

### Command Line Export Options

The export functionality can also be invoked from the command line, without using the graphical user interface. This can be convenient when regularly exporting data for several channels.

To see available options, run phoebus like this:

```
phoebus -main org.phoebus.archive.Export -help

Usage: -main org.phoebus.archive.Export [options]

Export archived data into files

General command-line options:

-help                                    -  This text
-settings settings.xml                   -  Import settings from file, either␣
↪exported XML or property file format

Archive Information options:
-list [pattern]                          -  List channel names, with optional pattern␣
↪('.', '*')

Data Export options:
-start '2019-01-02 08:00:00'             -  Start time
-end '2019-02-03 16:15:20'               -  End time (defaults to 'now')
-bin bin_count                           -  Export 'optimized' data for given bin␣
↪count
-linear HH:MM:SS                         -  Export linearly extrapolated data for␣
↪time intervals
-decimal precision                       -  Decimal format with given precision
-exponential precision                   -  Exponential format with given precision
-nostate                                 -  Do not include status/severity in tab-
↪separated file
-unixTimeStamp                           -  Use UNIX timestamp (in ms) instead of␣
↪formatted date/time string
-export /path/to/file channel <channels> -  Export data for one or more channels into␣
↪file

File names ending in *.m or *.mat generate Matlab files.
All other file name endings create tab-separated data files.
```

### 8.9.8 The Statistics-tab

Under the Statistics-tab, some basic statistical measures of the plotted data-points can be viewed.

It is important to note that the statistics are calculated only on the data values themselves *without taking into account the timestamps of data-points*: in the calculation of the statistical measures, only the value of data-points and the total number of data-points are taken into account, while neither the time interval under consideration (except indirectly for determining the subset of data-points to calculate the statistical measures for), nor the timestamps of individual data-points are part of the calculation.

For instance, in a plot based on both archived and live samples, the mean value will be skewed towards the live data portion if live data is sampled at a higher frequency than archived data.

## 8.10 Error Log

The Error Log is a read-only panel that copies and displays application messages. It simplifies checking "print" statements from scripts, or error messages that might explain why some display doesn't work as expected.

By default, these messages appear in the terminal where the Phoebus product was started. On MS Windows, however, there may not be a terminal. A site-specific launch script might also redirect the standard and error output of the product to a file. To the end user, it is thus not always obvious how to locate the application messages.

The "Error Log" shows messages written to the standard output, the error output or the console logger in one predictable place. It captures messages from the time it is started until it is closed.

The typical process to check for error message is thus:

1) Open the Error Log
2) Now open the display which is suspected of issuing a message
3) Check the Error Log, then close when no longer needed

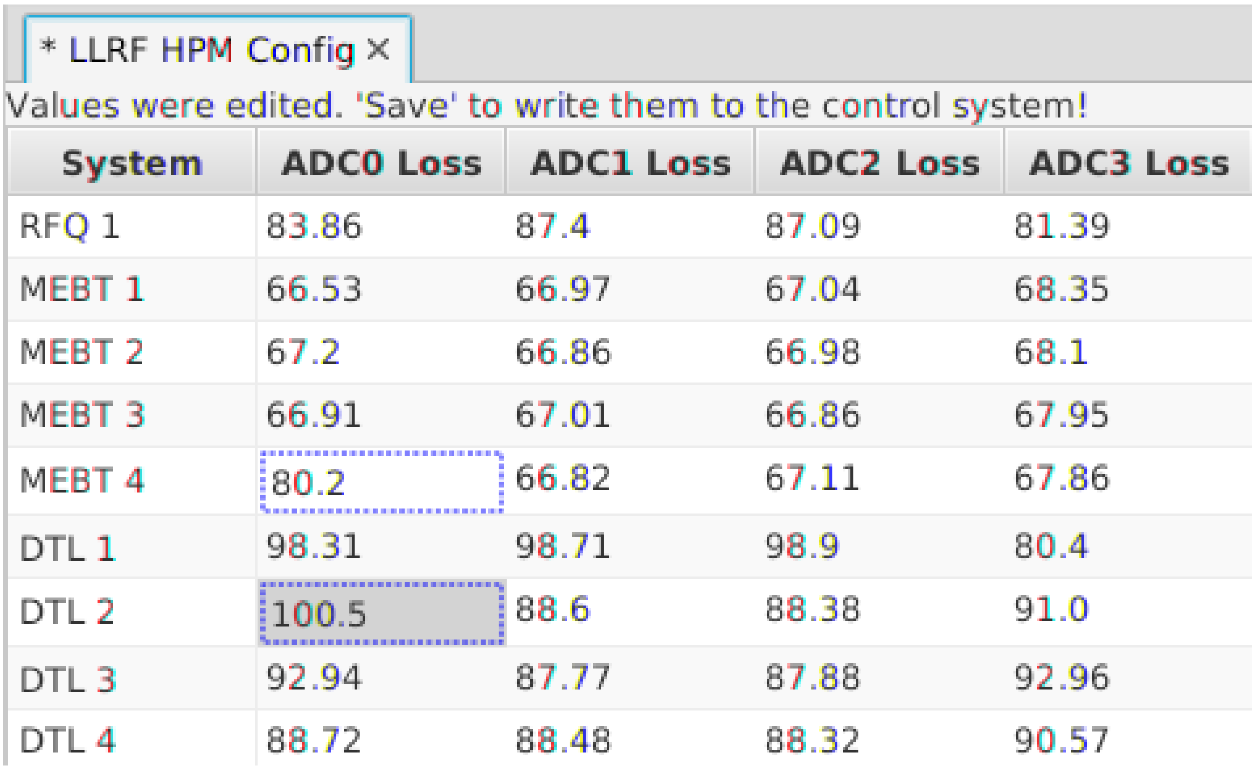
## 8.11 PACE: PV Adjust, Confirm, ELog

The PACE Editor provides a tabular view of Process Variables (PV) names and their current values.

The user can edit the values, review the changes, revert changes, and finally write changes back to the PVs with an automated electronic logbook (ELog) entry that lists the modified PVs and their values.

It is ideally suited to maintain critical setting PVs for a list of identical systems where each instance of a system has the same PVs, so these PVs can be shown in columns of a table. Changes to the PVs are documented via log entries.

The PV changes and the associated logbook entry are as much as possible treated as one transaction: PV writes are only attempted after a successful logbook entry. If the logbook entry fails, PVs are restored to their original value.



Values were edited. 'Save' to write them to the control system!

| System | ADC0 Loss | ADC1 Loss | ADC2 Loss | ADC3 Loss |
|---|---|---|---|---|
| RFQ 1 | 83.86 | 87.4 | 87.09 | 81.39 |
| MEBT 1 | 66.53 | 66.97 | 67.04 | 68.35 |
| MEBT 2 | 67.2 | 66.86 | 66.98 | 68.1 |
| MEBT 3 | 66.91 | 67.01 | 66.86 | 67.95 |
| MEBT 4 | 80.2 | 66.82 | 67.11 | 67.86 |
| DTL 1 | 98.31 | 98.71 | 98.9 | 80.4 |
| DTL 2 | 100.5 | 88.6 | 88.38 | 91.0 |
| DTL 3 | 92.94 | 87.77 | 87.88 | 92.96 |
| DTL 4 | 88.72 | 88.48 | 88.32 | 90.57 |

Fig. 1: PACE

### 8.11.1 Starting PACE

PACE is opened whenever a `*.pace` configuration file is opened, for example via a double-click in the file browser.

### 8.11.2 Using PACE

When a `*.pace` configuration file is opened, the table displays the current value of PVs. Except for columns marked as read-only in the configuration file (which is also indicated via the cell tool-tip), cell values can be edited.

Modified cells are indicated via a border.

The context menu allows restoring modified cells to their original value. The context menu can also be used to set several selected cells to the same value.

Finally, there are two options: You can commit the modified cell values to the respective PVs, or abandon the changes.

When trying to close a modified PACE table, a dialog will ask if you want to save the changes. Alternatively, you can also invoke the File/Save menu item.

A logbook submission dialog will appear, populated with information about the modified cells. When it is submitted, the PVs will be written.

To abandon the changes, simply close the PACE table and answer “No” when prompted if changes should be saved.

### 8.11.3 Configuration File

Configuration files in the `*.pace` format are typically prepared by control system personell, but end users may also create them. A PACE file uses an XML format with the following basic structure:

```
<paceconfig>
  <title> Your Title goes here  </title>
  <columns/>
  <instances/>
</paceconfig>
```

The columns section describes the columns of the PACE table. Each column has a name, optional access mode, and a macroized PV pattern:

```
<columns>
  <column>
    <name>Column Header</name>
    <access>rw</access>
    <pv>${S}_RCCS:CV${N}02:PID_KP</pv>
  </column>
  <!-- There can be more columns -->
</columns>
```

The cell access can be rw for read-write or ro for read-only. The PV name for the cell contains macros either written as `$(macro)` or `${macro}`. Replacement values for the macro variables are defined below under instances.

The column definition can have meta-PVs in addition to the main PV. These are meant to contain the data, user name and comment associated to the last change of the main PV. They are defined via tags similar to the main PV:

```
<!-- PV as before, but now followed by meta PVs:
<pv>${S}_RCCS:CV${N}02:PID_KP</pv>
<name_pv>${S}_RCCS:CV${N}02:PID_Name</name_pv>
```

(continues on next page)

(continued from previous page)

```
<date_pv>${S}_RCCS:CV${N}02:PID_Time</date_pv>
<comment_pv>${S}_RCCS:CV${N}02:PID_Txt</comment_pv>
```

The current values of these meta PVs will be displayed in the tool-tip of the cell. The name and date meta PVs will be updated with the user name and current date when writing to the main PV. The comment PV can be edited by providing access to it in a separate column.

Finally, each row in the PACE table is defined via instances:

```
<instances>
  <instance>
    <name>DTL 1</name>
    <macros>S=DTL,N=1</macros>
  </instance>
  <!--  Many more...: DTL 1..6, CCL 1..4. -->
</instances>
```

Each row starts with the name of that instance, followed by one cell for each column definition. The macro settings are used to turn the macro-ized PV names of the column definitions into specific PV instances.

A macro can be set to empty like this:

```
<macros>PV=""</macros>
```

If the PV name for a cell evaluates to an empty string, that cell will not have a PV.

Configuration File Example:

```
<!-- Example config file -->
<paceconfig>
  <title> Title </title>
  <columns>
    <column>
      <name>PID Gain</name>
      <access>rw</access>
      <pv>${S}_RCCS:CV${N}02:PID_KP</pv>
      <name_pv>${S}_RCCS:CV${N}02:PID_Name</name_pv>
      <date_pv>${S}_RCCS:CV${N}02:PID_Time</date_pv>
      <comment_pv>${S}_RCCS:CV${N}02:PID_Txt</comment_pv>
    </column>
    <column>
      <name>Comment</name>
      <access>rw</access>
      <pv>${S}_RCCS:CV${N}02:PID_Txt</pv>
    </column>
    <!-- There can be more columns -->
  </columns>
  <instances>
    <instance>
      <name>DTL 1</name>
      <macros>S=DTL,N=1</macros>
    </instance>
    <!--  Many more...: DTL 1..6, CCL 1..4. -->
  </instances>
</paceconfig>
```

## 8.12 PV Tree

The PV Tree displays the hierarchical data flow between EPICS records. It displays the record types and their current values as well as severity/status. It attempts to reflect the data flow by traversing input links (INP, INPA, DOL, . . . ).

The PV Tree has two modes:

**"Run"** **:** In this mode it will always display the current value of each item in the tree.

**"Freeze on Alarm"** **:** In this mode, updates pause as soon as the root item of the PV Tree goes into alarm.

### 8.12.1 Usage

Enter a name into the "PV" text box, and see what happens.

### 8.12.2 Tool Bar Buttons

, Changes the PV Tree mode between "running" and "freeze on alarm".

Collapse the tree, i.e. close all sub-sections of the tree.

Display all items in the tree that are in an alarm state. Note that this is performed whenever you push the tool bar button. If the PV tree items update, branches will not automatically show or hide based on their alarm state, because this could result in a very nervous display for a rapidly changing PV tree. Whenever you desire to update the tree to show/hide items, push the button.

Expand all sub-sections of the tree.

### 8.12.3 Limitations

This tool uses the EPICS Channel Access or PV Access network protocols to read PVs. Note that there is a difference between EPICS records in an IOC and channels on the network. There is no way to query EPICS IOCs for their database information to determine the available "input" links.

Given a PV name `x`, the PV tree attempts to read the channel `x.RTYP`. If the channel is indeed based on a record, it will report the record type. The knowledge of which links to follow for each record type is configured into the EPICS PV Tree application via the `org.phoebus.applications.pvtree/fields` preference setting. This allows maintainers of site-specific settings to add support for locally developed record types, or add certain output links to the list of links that the PV tree should trace and display.

The Channel Access protocol adds another limitation to the PV tree operation, because Channel Access strings are restricted to a length of 40 characters. The PV tree can therefore not read the complete value of links when they exceed 40 characters. This results in long record names being truncated and then failing to resolve. As a workaround, the PV tree can read a link `x.INP` as `x.INP$` with a trailing dollar sign, which causes the IOC to return the value as a byte waveform without length limitations. This mode, however, is not supported by older IOCs and older CA gateways. If your site only runs IOCs and gateways that support the `x.INP$` channel name syntax, you can enable the `org.phoebus.applications.pvtree/read_long_fields=true` option in the PV tree preferences. If your site still runs older IOCs, you won't be able to use the PV tree with them unless you set `org.phoebus.applications.pvtree/read_long_fields=false`.

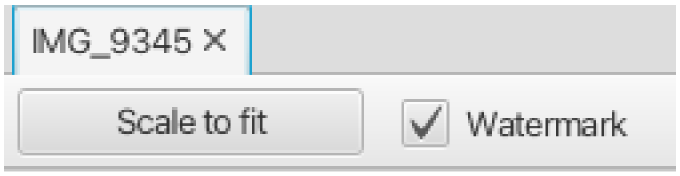
## 8.13 Image Viewer

### 8.13.1 Overview

As the name suggests, the Image Viewer renders an image file at full resolution in a separate tab in the main window. The tab title shows the image file name.

Using the "Scale to fit" button user may rescale the images to fit to the current size of the view. The resizing process will honor the aspect ratio of the image, i.e. the resulting view may show empty space above/below the image, or to left/right of the image. Pressing the button again will switch back to showing the image at full resolution. It should be noted that if the image is smaller than the current size of the view, pressing the "Scale to fit" button will zoom in on the image.

In some cases it may be necessary to indicate that the image is static in nature and cannot be interacted with, e.g. when showing a screen shot of an OPI. Ticking the Watermark check box will add an overlay showing the text "WATERMARK" (may be customized).



There is no menu entry for the application. It is launched automatically for instance when user double clicks on an image file in the File Browser, or when user clicks on an image in the log entry attachment viewer.

Supported file types are png, jpg (jpeg) and gif. The application also supports Scalable Vector Graphics files, but user should keep in mind that SVG tools may generate incompatible files. When a SVG file is rendered by the application the "Scale to fit" button is disabled.

If user has configured an external application for a particular image file extension, this is NOT overridden by the Image Viewer application.

## 8.14 Credentials Management

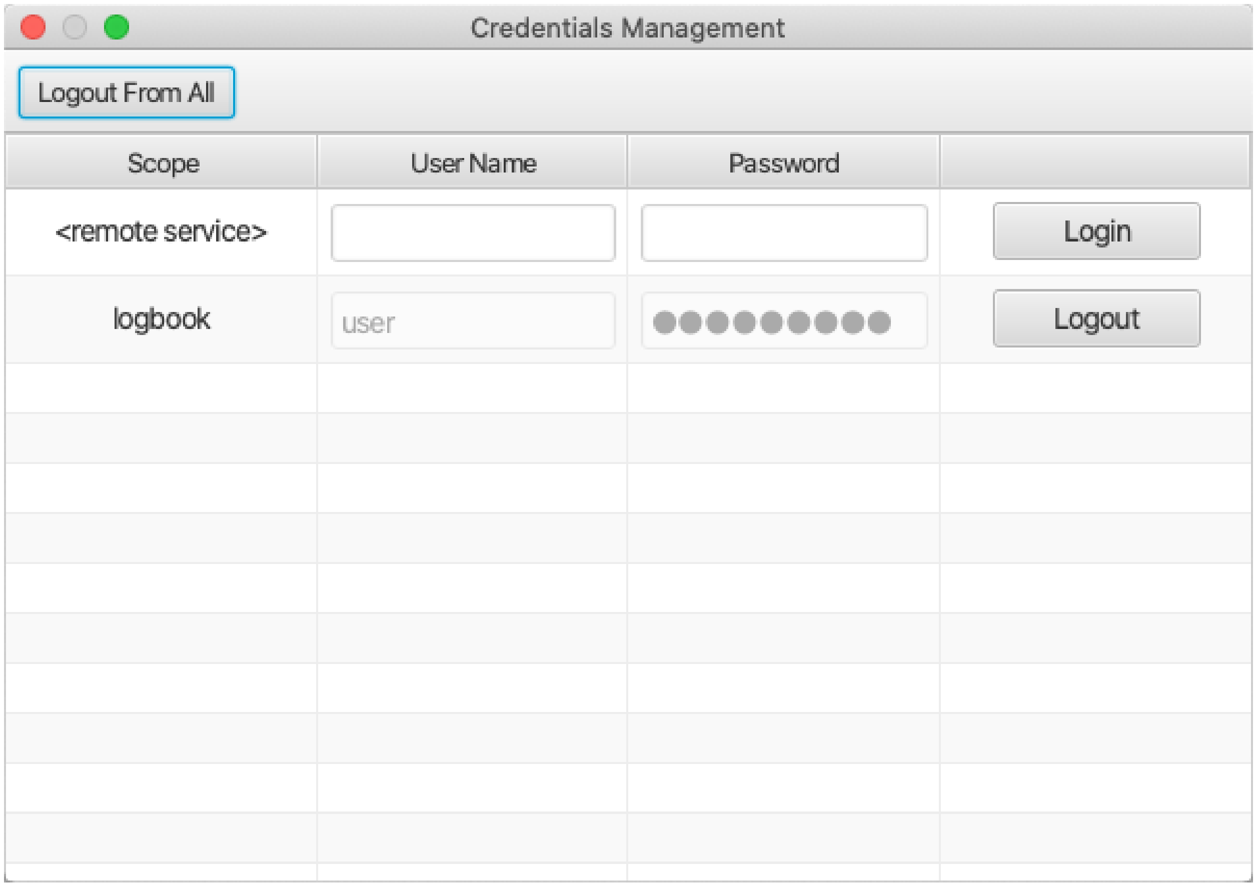
Some applications may need to prompt the user for credentials, e.g. when interacting with a protected remote service. One such example would be the electronic logbook.

Further, Phoebus may be configured to store the credentials entered by the user to avoid repeated prompts. In order to also support an explicit logout capability, the Credentials Management application offers means to remove stored credentials.

In some cases an explicit login procedure can be useful, e.g. login to service for the purpose of storing user credentials and thereby support automated creation of logbook entries.

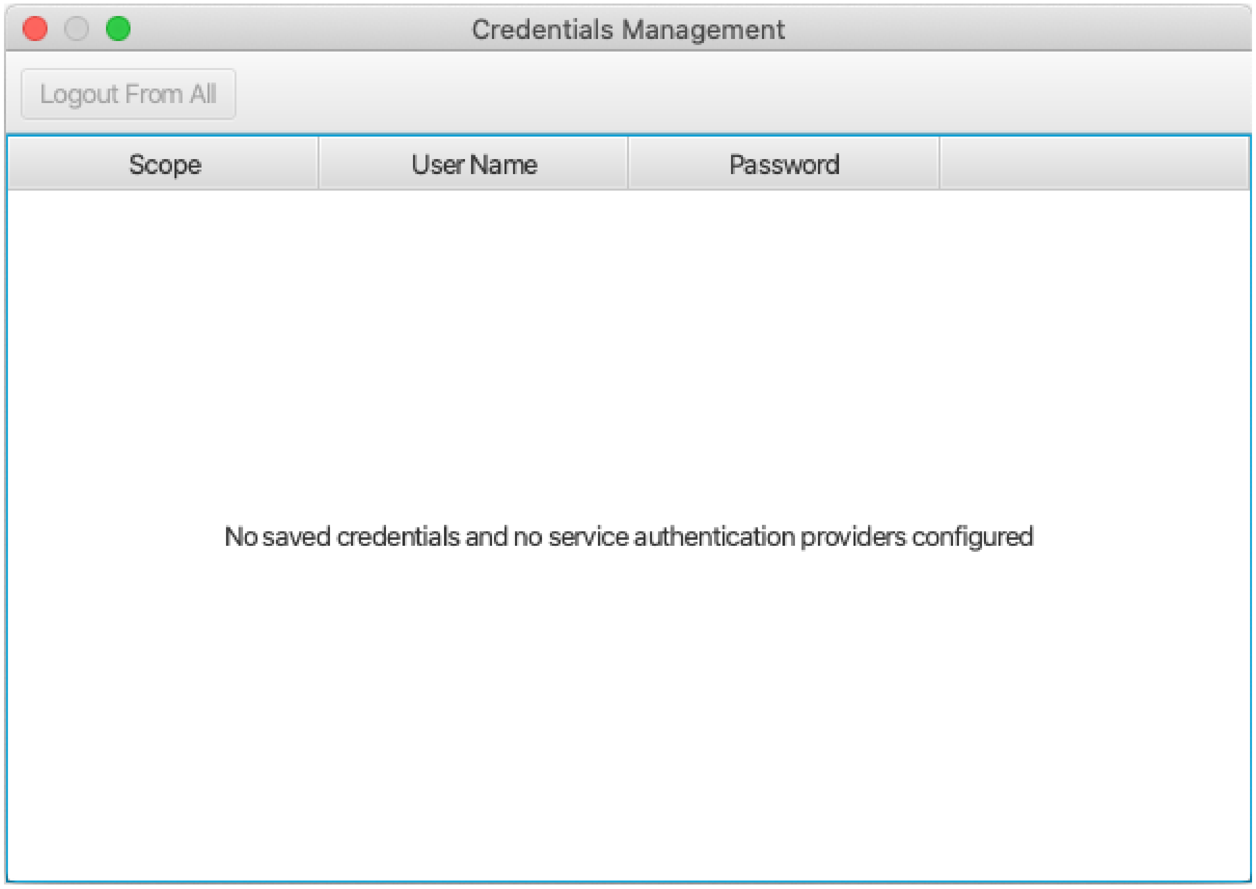
The application is launched using the dedicated button in the (bottom) status bar.

The below screen shot shows an example where credentials have been stored for the "logbook" scope, plus an option to login to the "<remote service>" scope. User may also choose to "logout" from all scopes, i.e. to remove all stored credentials.



| Scope | User Name | Password | |
|---|---|---|---|
| <remote service> | | | Login |
| logbook | user | ●●●●●●●●● | Logout |

If no credentials are stored in the credentials store, and if no services supporting authentication have been configured, the Credentials Management UI will show a static message:



No saved credentials and no service authentication providers configured

## 8.15 Channel Applications

### 8.15.1 Overview

The Channel Viewer is a CS-Studio application which can query the directory service for a list of channels that match certain conditions, such as physical functionality or location. It also provides mechanisms to create channel name aliases, allowing for different perspectives of the same set of channel names.

### 8.15.2 Launching

From within cs-studio `Applications --> Channel --> Channel Table/Channel Tree`

From command line

`-resource cf://?query=SR*&app=channel_tree` `-resource cf://?query=SR*&app=channel_table`

### 8.15.3 Channel Table

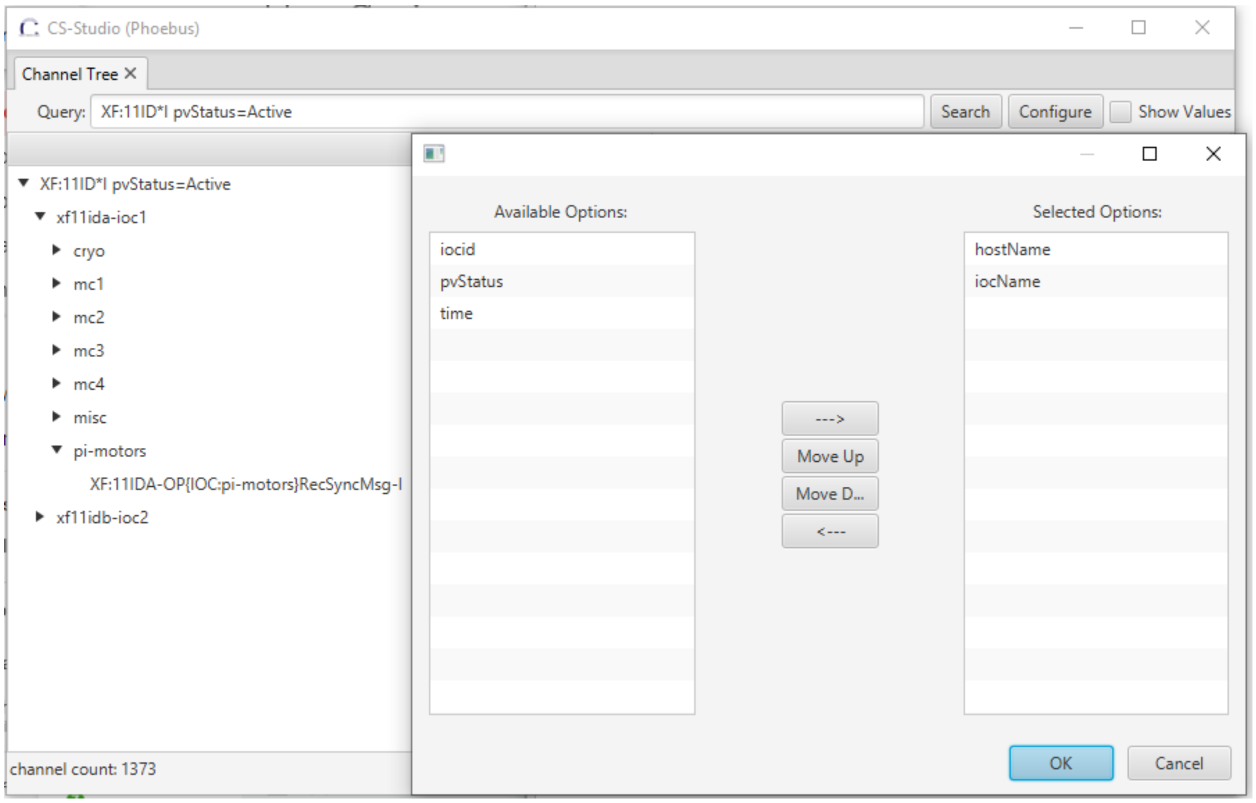
Displays the results of a channelfinder query as a table

The query can be based on the channel name or based on a group of tag and properties associated with the channel

Wildcard character like "*", "?" can be used in the queries

### 8.15.4 Channel Tree

Channel Tree by Property allows to create an hierarchical view of the channels by using properties and their values. It groups the channels returned by a query based on the value of the properties selected.



Based on the order of the selected properties, all channels with a same property value are grouped under the same node. In the image above we see the channels ordered by the value of the properties `hostName` and then `iocName`

## 8.16 Display Builder Converters

### 8.16.1 Converting BOY `*.opi` Displays

The display builder can read existing BOY `*.opi` displays. It will automatically convert the widgets and their properties from the legacy file format, so you can simply open exiting `*.opi` displays in the display builder runtime.

Since there can be subtle differences in the look and behavior of the widgets, you may want to manually adjust a few displays. Do this by opening the existing `*.opi` in the Display Builder editor, adjust as necessary, and then save the file with the new `*.bob` file extension. When the display builder runtime opens a `*.opi` display, it will check for the existence of a `*.bob` version and open the latter. There is thus no need to delete the older `*.opi` files, especially as you transition between tools. Most `*.opi` files will work "as is", without changes, in both BOY and Display Builder. You only need to create slightly adjusted `*.bob` file versions as necessary, and the Display Builder runtime will open these instead of the `*.opi` files.

Manual adjustments will be necessary for displays that use plots or scripts. The Display Builder offers a different set of plots, so legacy displays need to be updated. Since the underlying Java API of the two tools is dramatically different, any scripts beyond this will need to be rewritten:

```
value = PVUtil.getDouble(pvs[0])
widget.setPropertyValue("x", value + 100)
```

See `script_util/portable.py` in the examples for hints.

If you prefer to bulk-convert existing `*.opi` files into the new file format, you can do that via this command line invocation:

```
Usage: phoebus -main org.csstudio.display.builder.model.Converter [-help] [-output /
↪path/to/folder] <files>

Converts BOY *.opi files to Display Builder *.bob format

-output /path/to/folder   - Folder into which converted files are written
<files>                   - One or more files to convert
```

### 8.16.2 Converting MEDM `*.adl` Displays

When you open an MEDM `*.adl` in the Phoebus file browser, it converts this into a `*.bob` file which you can then open in the Display Builder editor or runtime.

For bulk conversions, you can use this command line invocation:

```
Usage: phoebus -main org.csstudio.display.converter.medm.Converter [-help] [-output /
↪path/to/folder] <files>

Converts MEDM *.adl files to Display Builder *.bob format

-output /path/to/folder   - Folder into which converted files are written
<files>                   - One or more files to convert
```

### 8.16.3 Converting EDM `*.edl` Displays

To use the EDM converter, add the following to your settings:

```
org.csstudio.display.converter.edm/edm_paths_config=/path/to/edm_paths.txt
org.csstudio.display.converter.edm/colors_list=/path/to/edm/colors.list
```

For details, see full description of *Preferences Listing*.

Each time you now try to open an `*.edl` file, the converter will create a `*.bob` file in the same location and then open it.

For bulk conversions, you can use this command line invocation, which can convert a list of files, including complete directories:

```
Usage: -main org.csstudio.display.converter.edm.Converter [options] <files>

Converts EDM *.edl files to Display Builder *.bob format.

Options:
-help                          - Help
-colors /path/to/colors.list - EDM colors.list file to use
-paths /path/to/paths.list    - File that lists paths
-output /path/to/folder       - Folder into which converted files are written
```

(continues on next page)

(continued from previous page)

```
-force                          - Overwrite existing files instead of stopping
-depth count                    - Convert just the listed files (1), or also referenced␣
↪files (2), or more levels down
```

The batch converter can also recursively convert referenced files like embedded displays or related displays. Refer to the complete `-help` output for details.

### 8.16.4 Auto-Converter

When a user opens an `*.edl` file, it is automatically converted into a `*.bob` file and the latter then opens.

Furthermore, there is an EDM auto-converter that can automatically look for `*.edl` files and convert them. This way, a site that plans to slowly transition from EDM to the Display Builder does not need to bulk-convert all files. Instead, you start with a top-level display that for example includes links to displays `a.bob` and `b.bob`. When the user then tries to open `a.bob` and it does not exist, the auto-converter will search for a file `a.edl` and auto-convert it. The next time around, the `a.bob` file exists and opens a little faster. This way, files are auto-converted on first access, on-demand.

To enable the auto-converter, define a folder where the converted files will be stored as well as related settings:

```
org.csstudio.display.converter.edm/auto_converter_dir=/path/to/AUTOCONVERTED_FILES
org.csstudio.display.converter.edm/auto_converter_strip=/some/prefix/to/strip
org.csstudio.display.converter.edm/edm_paths_config=/path/to/my_edm_search_paths.txt
```

With the auto-converter folder defined, each time the display builder needs to open a file `*.bob` that does not exist, it will remove the `auto_converter_strip` prefix, and try to locate an `*.edl` file of that name along the `edm_paths_config`. If an EDM file is found, it is converted and written to the `auto_converter_dir`. In case the EDM file is found via an http link, it is first downloaded. On success, the resulting `*.bob` file is opened. When that display then refers to other EDM files, the same process is repeated. Converted files are stored in the `auto_converter_dir`, they are thus only fetched and converted once.

As an example, assume EDM files are located on a web server under `https://my.site.org/opi/edm` and you want to start by opening `https://my.site.org/opi/edm/general/start.edl`.

Use these example settings:

```
org.csstudio.display.converter.edm/auto_converter_dir=$(user.home)/AUTOCONVERTED_FILES
org.csstudio.display.converter.edm/auto_converter_strip=$(user.home)/AUTOCONVERTED_
↪FILES
org.csstudio.display.converter.edm/edm_paths_config=https://my.site.org/opi/edm/paths.
↪txt
```

where the file `paths.txt` on the server should include just one line:

```
https://my.site.org/opi/edm/
```

To bootstrap access to the EDM displays from your display builder screens, use an action button labeled "EDM Displays" with an action to open `$(user.home)/AUTOCONVERTED_FILES/general/start.bob`. When you first click that button, `$(user.home)/AUTOCONVERTED_FILES/general/start.bob` does not exist, and we attempt to auto-convert it from an EDM display:

- The `auto_converter_strip` prefix is removed, leaving `general/start.bob`
- Using the search path listed in the file provided by `edm_paths_config`, the corresponding EDM file is found as `https://my.site.org/opi/edm/general/start.edl`

- It is downloaded as `$(user.home)/AUTOCONVERTED_FILES/general/start.edl` and converted into `$(user.home)/AUTOCONVERTED_FILES/general/start.bob`

From now on, `$(user.home)/AUTOCONVERTED_FILES/general/start.bob` exists and simply opens right away.

## 8.17 Display Navigation

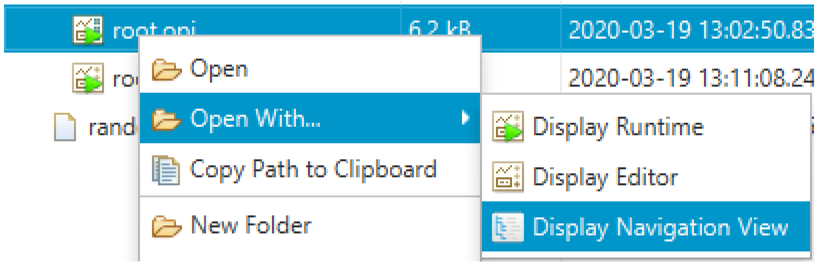
A visual representation of all the navigation graph of an OPI screen.

The application searches all the controls of an OPI screens for actions which launch other OPI screens. The information is then formatted into a

1. A unique list of all the OPI screens which are linked to the current OPI screen.
2. A tree view showing the linked files while in a manner which preserves the

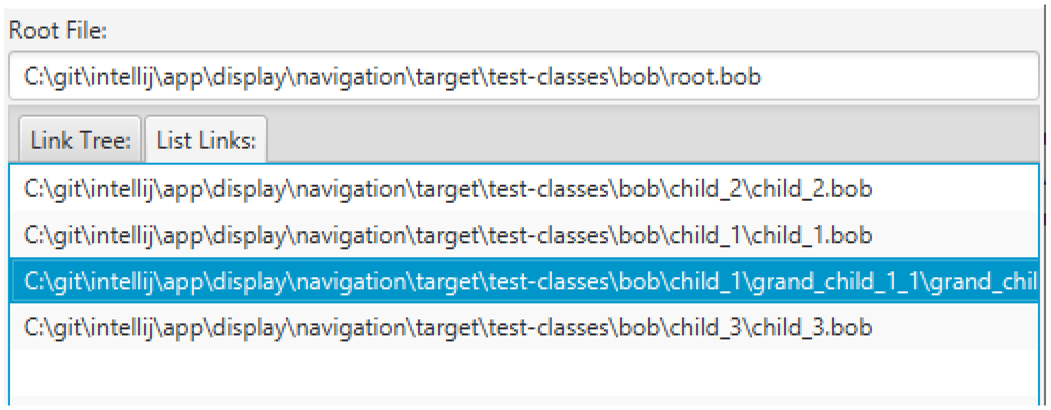
### 8.17.1 Opening the Display Navigation View

The context menu for .opi and .bob files will have an action Open with *Display Navigation View*



### 8.17.2 List Link

The list link view shows a unique of all the OPI files linked to the selected OPI. Loops and multiple links are ignored to create a flat list of only unique OPI screens.


Root File:

C:\git\intellij\app\display\navigation\target\test-classes\bob\root.bob

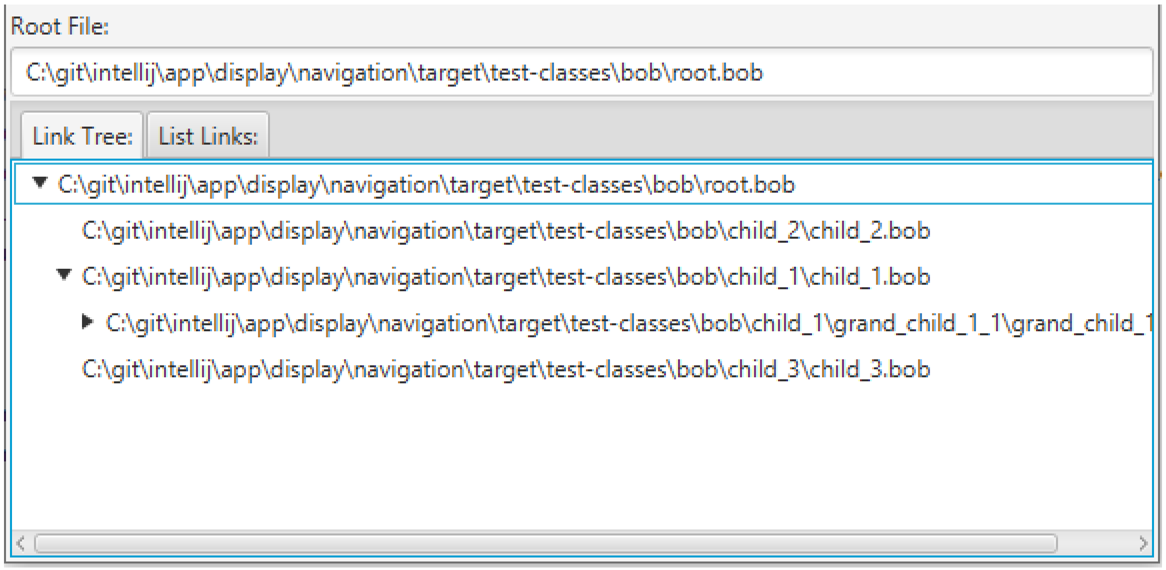
Link Tree: List Links:

C:\git\intellij\app\display\navigation\target\test-classes\bob\child_2\child_2.bob
C:\git\intellij\app\display\navigation\target\test-classes\bob\child_1\child_1.bob
C:\git\intellij\app\display\navigation\target\test-classes\bob\child_1\grand_child_1_1\grand_chil
C:\git\intellij\app\display\navigation\target\test-classes\bob\child_3\child_3.bob

The *copy* context menu action can be used to copy the complete path of all linked files.

### 8.17.3 Link Tree

A tree view of all linked files to the current OPI screen. The tree view does not break loops and cyclic links.


Root File:

C:\git\intellij\app\display\navigation\target\test-classes\bob\root.bob

```
▼ C:\git\intellij\app\display\navigation\target\test-classes\bob\root.bob
    C:\git\intellij\app\display\navigation\target\test-classes\bob\child_2\child_2.bob
  ▼ C:\git\intellij\app\display\navigation\target\test-classes\bob\child_1\child_1.bob
    ▶ C:\git\intellij\app\display\navigation\target\test-classes\bob\child_1\grand_child_1_1\grand_child_1
    C:\git\intellij\app\display\navigation\target\test-classes\bob\child_3\child_3.bob
```

## 8.18 Display Builder

Display Builder is an editor and a runtime for build and running controls GUIs

### 8.18.1 Create new Display

Use the menu option `Applications`, `Display`, `New Display` to create a new display.

### 8.18.2 Install Examples

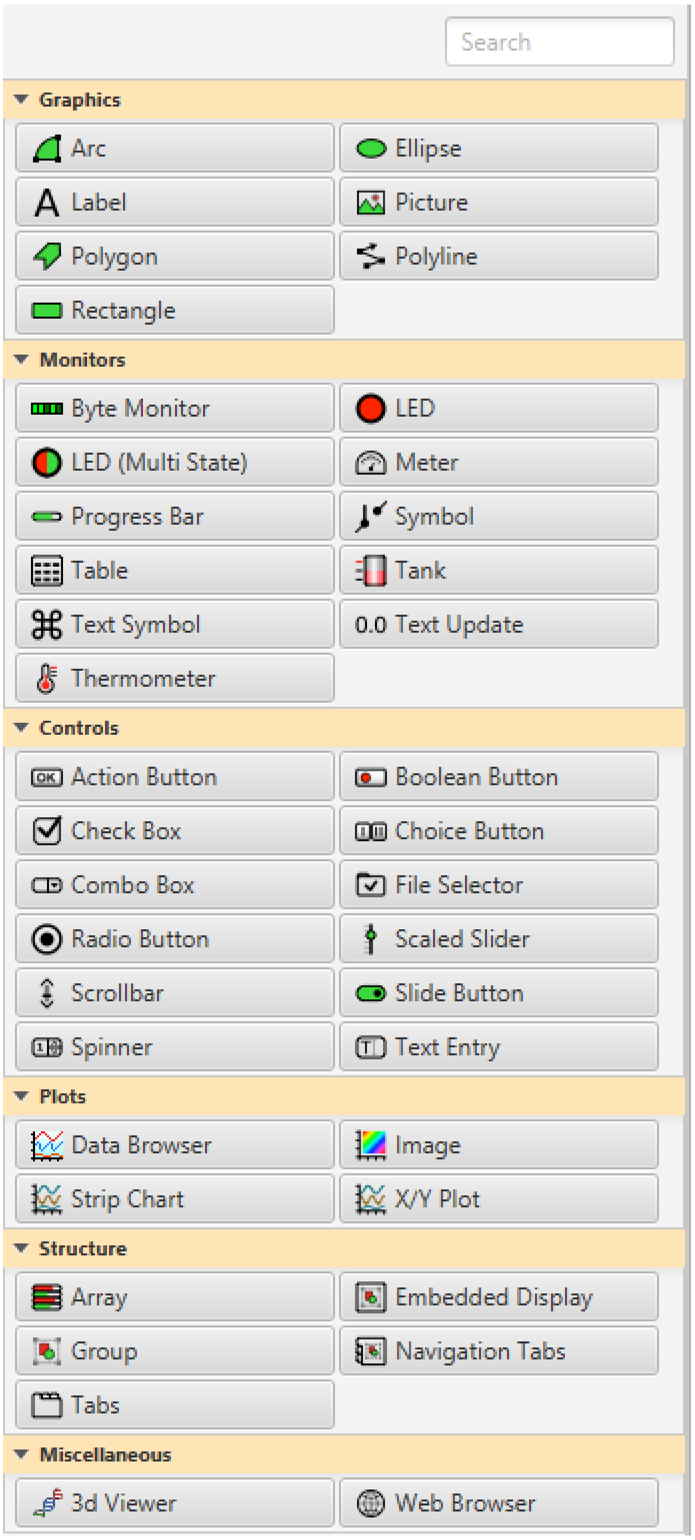
Invoke `Install Example Display` from the menu `Applications`, `Display`, `Examples` to install the display builder examples.

You will be prompted for a location where you wish to install them.

Then use the menu `File`, `Open` to open the file `01_main.bob` from the examples to get started.

### 8.18.3 Widgets

The Display Builder application supports a set of widgets which can be used to create control GUIs. The widgets are organized based on functionality as shown below



### Widget Properties

Each widget has a set of properties which describe the widgets visual and functional behaviours.

#### Common Widget Properties

**The following most basic properties are available on all widgets:**

- "x", "y", "width", "height" that define its position within the display.
- "name" that can be used to identify a widget.
- "tooltip", used to provide relevant information when mouse pointer hovers over the widget. See below for additional information.

**While not available on all widgets the following properties are associated with almost all widgets**

- "foreground_color"
- "font" the font associated with the text of the widget
- "pv_name" the name of a datasource channel which is to be monitored and the values from the channel are used to update the state of the widget

#### Tool Tip Property

Every widget supports the tool tip property, but the default value set in the display editor differs among widgets. For PV aware widgets the default value is `$(pv_name)\n$(pv_value)`. In case a PV is not defined by the user for such a widget, the macros `$(pv_name)` and `$(pv_value)` will not expand to anything useful. In such cases user should consider to change the tool tip property value, or set it to an empty value.

An empty tool tip property will suppress rendering of a widget tool tip, even if the value for the tool tip text is set by a rule.

#### Widget Classes

Display builder allows the creation of "classes" of widget. A widget class is a way of defining a widget with a set of pre defined property values. This can be very useful in developing GUI standards and enforcing them.

For details look at the "using classes" section in the examples: *Install Examples*

### 8.18.4 Datasources

### 8.18.5 Macros

Macros can be used for string replacement in a variety of use cases. A common use case is to define macros as substrings of PV names used in displays controlling multiple instances of the same type of hardware.

The macro format can be either **$(macro_name)** or **${macro_name}**. Letters and numbers can be used for macro names, but a macro name must must start with a letter. Macro names are case sensitive.

There are a number of different macro types:

#### Symbol Macros

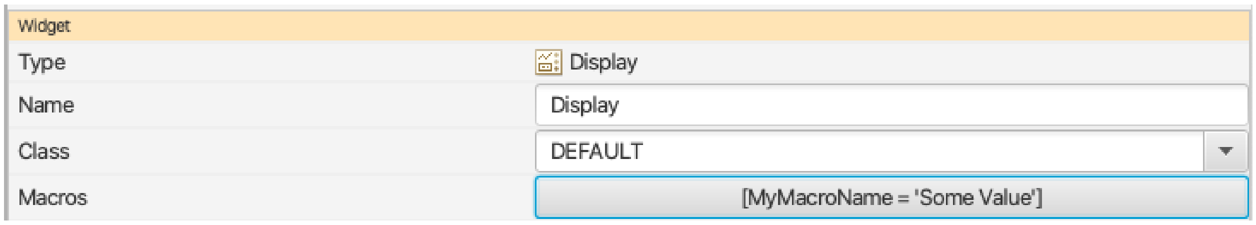
Symbol Macros can be defined in several places.

1. On the command line when launching a display using the -resource option:

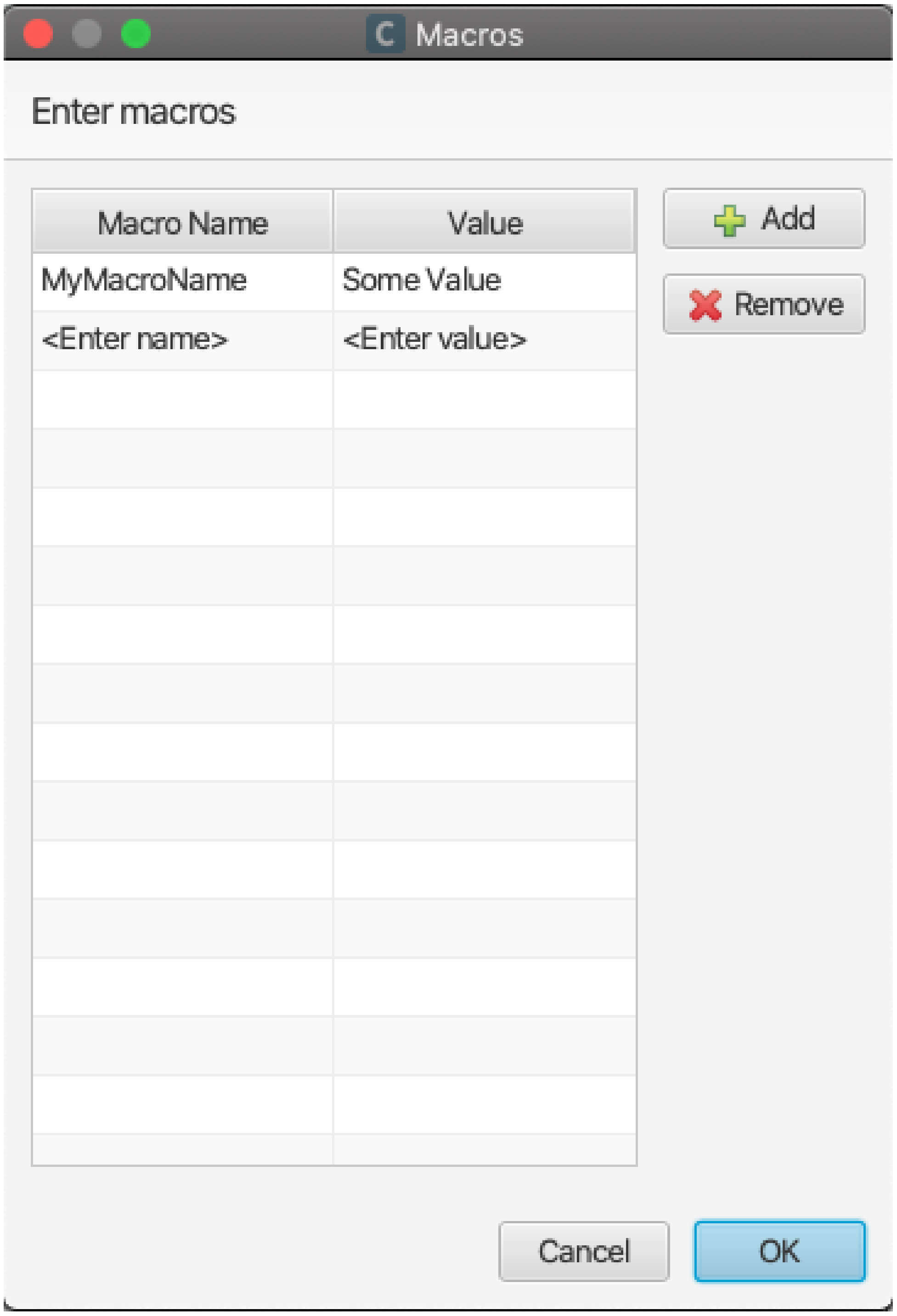
```
java -jar /path/to/application.jar -resource file:///path/to/display.bob?
↪MyMacroValue=Some%20Value
```

Note that macro values must be URL encoded, e.g. space character is encoded as %20.

2. In a display file using the macro property editor:



| Widget | |
|---|---|
| Type | Display |
| Name | Display |
| Class | DEFAULT |
| Macros | [MyMacroName = 'Some Value'] |



| Macro Name | Value |
|---|---|
| MyMacroName | Some Value |
| <Enter name> | <Enter value> |

3. A macro property editor is also available for container widgets like for instance Embedded Display, Group or Navigation Tabs.

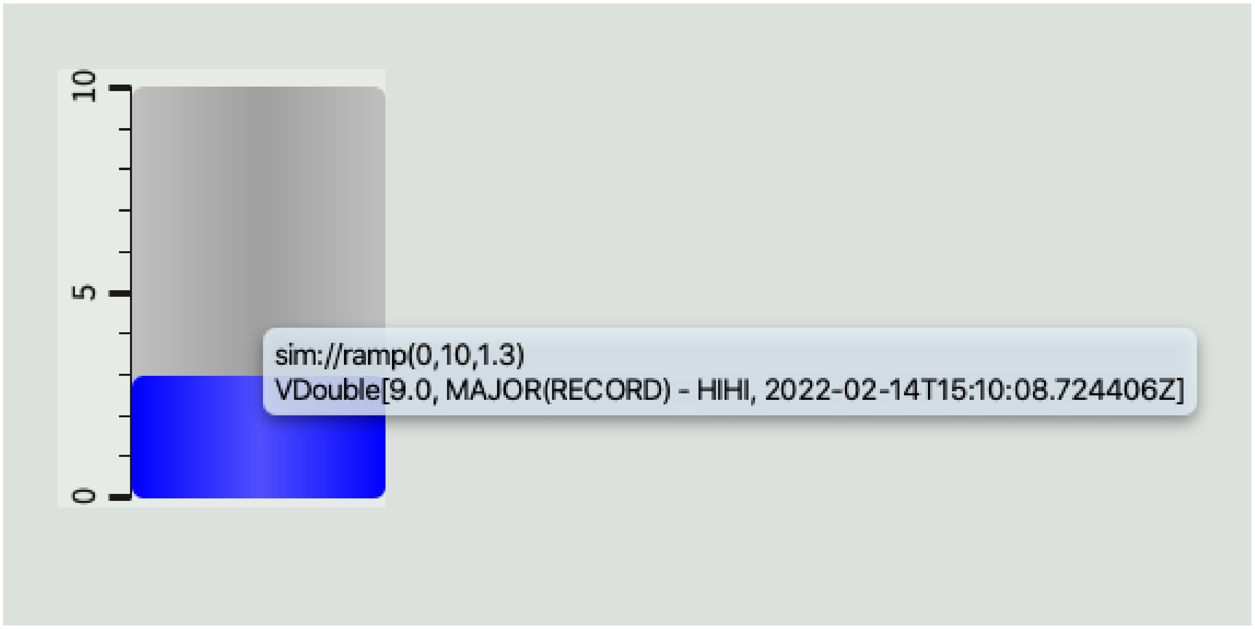
While macros will in general propagate (e.g. from command line to Embedded Display), overrides must be considered if the same macro - identified by name - is defined on multiple levels:

1. Macro defined in a display file will override macro defined on command line.
2. Macro defined for a container widget will override macro defined in the containing display file.

### Widget Property Value Macro

This allows you to access the value of a property of the widget in run mode. In this case, the macro_name is the property id of the widget property. For example, $(pv_name), $(pv_value), $(foreground_color).

A good example of Widget Property Value Macro is the tooltip: A tooltip of "$(pv_name)$(pv_value)" will display the PV Name and its value in run mode.



```
sim://ramp(0,10,1.3)
VDouble[9.0, MAJOR(RECORD) - HIHI, 2022-02-14T15:10:08.724406Z]
```

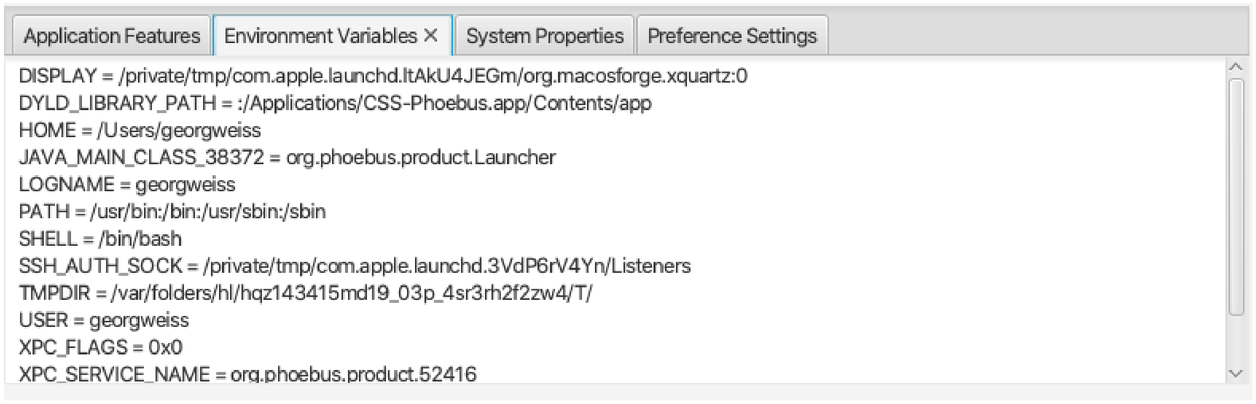
### System Macros

The following macros are predefined, and can be overridden by Symbol Macros:

- $(DID): The unique ID for each display. Even if there are multiple displays refering to the same physical OPI file, each display still has an unique ID. This macro is useful for avoiding name conflict. For example, if we add a prefix of $(DID)_ to a local PV name, we can easily guarantee the PV name is unique to this display.
- $(DNAME): The name of the display.

### Environment Variables

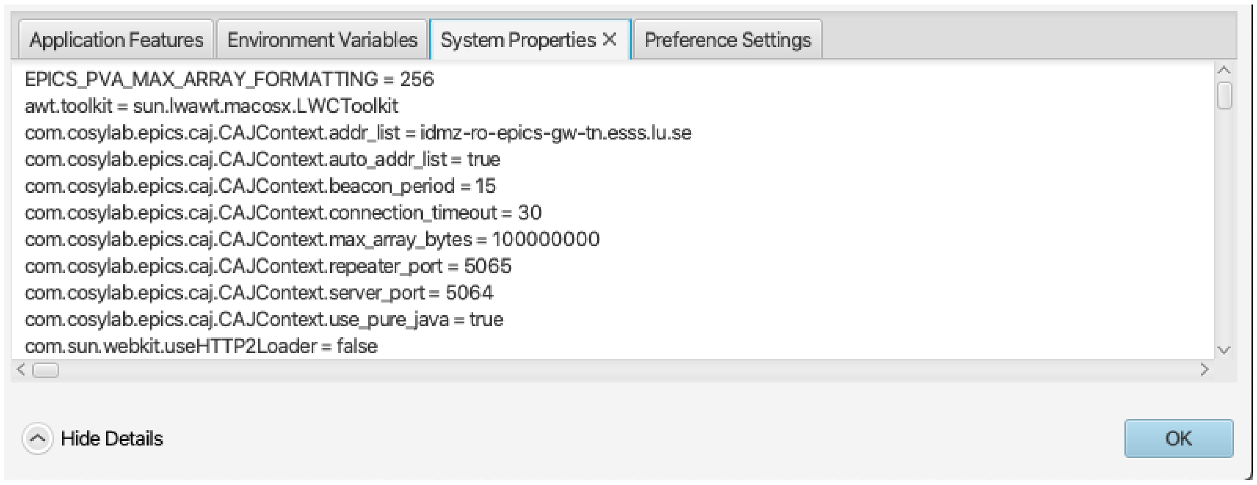
A macro like $(PATH) will - unless explicitly overridden - expand to the value of the process owner's environment variable PATH. To see the list of available environment variables, select menu option *Window->About* and then expand *Show Details* and select *Environment Variables* tab:



```
DISPLAY = /private/tmp/com.apple.launchd.ItAkU4JEGm/org.macosforge.xquartz:0
DYLD_LIBRARY_PATH = :/Applications/CSS-Phoebus.app/Contents/app
HOME = /Users/georgweiss
JAVA_MAIN_CLASS_38372 = org.phoebus.product.Launcher
LOGNAME = georgweiss
PATH = /usr/bin:/bin:/usr/sbin:/sbin
SHELL = /bin/bash
SSH_AUTH_SOCK = /private/tmp/com.apple.launchd.3VdP6rV4Yn/Listeners
TMPDIR = /var/folders/hl/hqz143415md19_03p_4sr3rh2f2zw4/T/
USER = georgweiss
XPC_FLAGS = 0x0
XPC_SERVICE_NAME = org.phoebus.product.52416
```

### System Properties

Java system properties can be accessed as macros, e.g. $(os.version). The list of supported Java system properties may vary between Java versions. To see the list of available system properties, both those defined by the Java runtime and those defined by the application, select menu option *Window->About* and then expand *Show Details* and select *System Properties* tab:



```
EPICS_PVA_MAX_ARRAY_FORMATTING = 256
awt.toolkit = sun.lwawt.macosx.LWCToolkit
com.cosylab.epics.caj.CAJContext.addr_list = idmz-ro-epics-gw-tn.esss.lu.se
com.cosylab.epics.caj.CAJContext.auto_addr_list = true
com.cosylab.epics.caj.CAJContext.beacon_period = 15
com.cosylab.epics.caj.CAJContext.connection_timeout = 30
com.cosylab.epics.caj.CAJContext.max_array_bytes = 100000000
com.cosylab.epics.caj.CAJContext.repeater_port = 5065
com.cosylab.epics.caj.CAJContext.server_port = 5064
com.cosylab.epics.caj.CAJContext.use_pure_java = true
com.sun.webkit.useHTTP2Loader = false
```

### Default Values

When using a macro as in **$(macro_name)** and the value for **macro_name** is not defined, the result will be an empty string. The syntax **$(macro_name=default_value)** can be used to yield the text **default_value** unless **macro_name** has a defined value.

One use case for default macro values are displays that allow standalone tests. When a display with **$(pv=sim://sine)** is executed with a value for the **pv** macro, that value will be used, but the display can also be opened "standalone" and will then use the default value of **sim://sine**.

### General Remark

A macro is a string substitution mechanism, nothing else. In particular, a macro contains no type information. This has implications if macros are referenced in rules. For instance, if compared to a string value, the macro must be quoted. On the other hand, if compared to a numerical value, the macro must expand to a number and be specified without quotes.

### 8.18.6 Dynamic

There are instances when setting static values for widgets properties is not enough. There are some use cases where one would like to change the visibility, color, and some other attributed of a widget dynamically at runtime.

The Display builder provides three mechanism to process data from one or more datasources at runtime and dynamically configure some aspects of a widget.

Care should be taken when using these mechanism, incorrect usage of these can have an adverse impact of the performance and behaviour of Phoebus.

#### Formula Functions

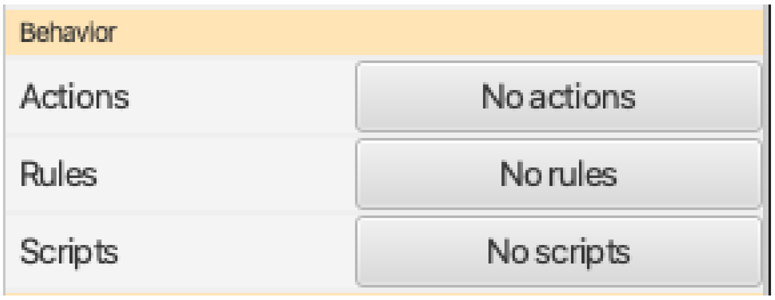
#### Rules

With rules you can change widget properties dynamically. A widget property value will change along with the boolean expression status or input PV value. The execution of a rule is triggered by its input PV(s), so at least one trigger PV is needed for a rule.

For more complex dynamic behavior you will need to use scripts.

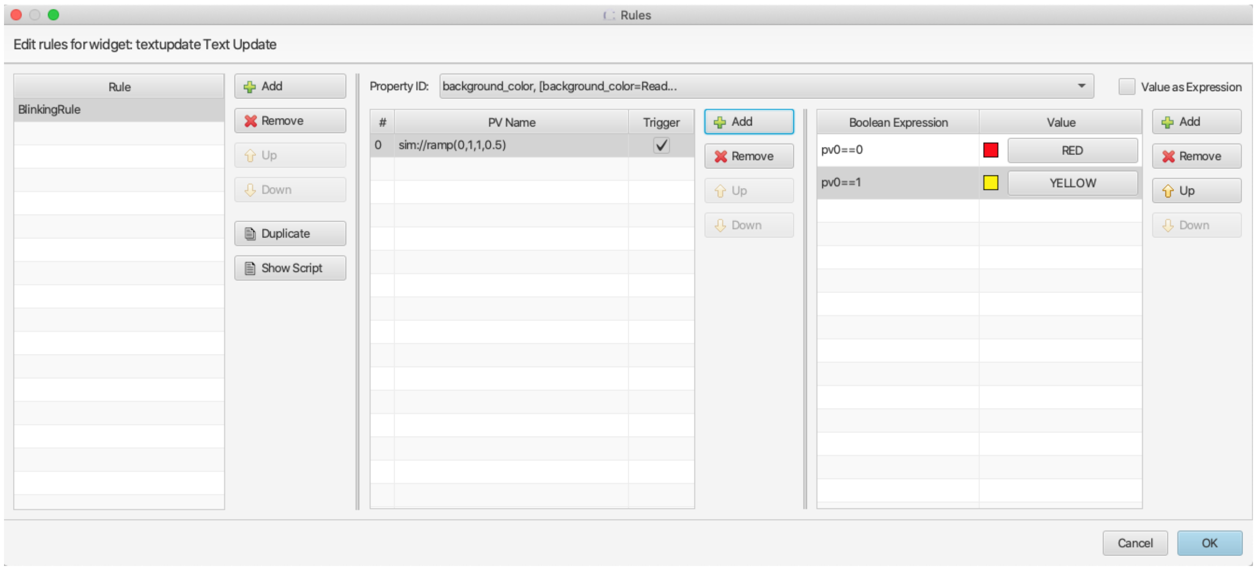
#### Attaching rules to a widget

1. In the widget properties pane, click the button to launch the rules editor dialog:



| Behavior | |
|---|---|
| Actions | No actions |
| Rules | No rules |
| Scripts | No scripts |

2. In the rules editor you can add one or multiple rules for a widget.



Edit rules for widget: textupdate Text Update

| Rule |
|---|
| BlinkingRule |

Property ID: background_color, [background_color=Read...

Value as Expression: ☐

| # | PV Name | Trigger |
|---|---|---|
| 0 | sim://ramp(0,1,1,0.5) | ✓ |

| Boolean Expression | | Value |
|---|---|---|
| pv0==0 | 🟥 | RED |
| pv0==1 | 🟨 | YELLOW |

- A rule must have a name, which defaults to “New Rule”.
- The “Property ID” drop-down will list the widget properties that may be controlled by rules.
- The middle section lists the input PVs for the rule selected in the rule list. The check box in the “Trigger” column determines if a value change in the PV should apply the rule.
- The right-most list holds the boolean expressions and values of the widget property. When a “Boolean Expression” evaluates to `true`, the widget property selected in the “Property ID” drop-down will be set to the value defined in the “Value” column.
- The “Show Script” button will display the script that is constructed from the rule. Any changes to the script content are ignored.
- The “Value as Expression” checkbox is explained below.

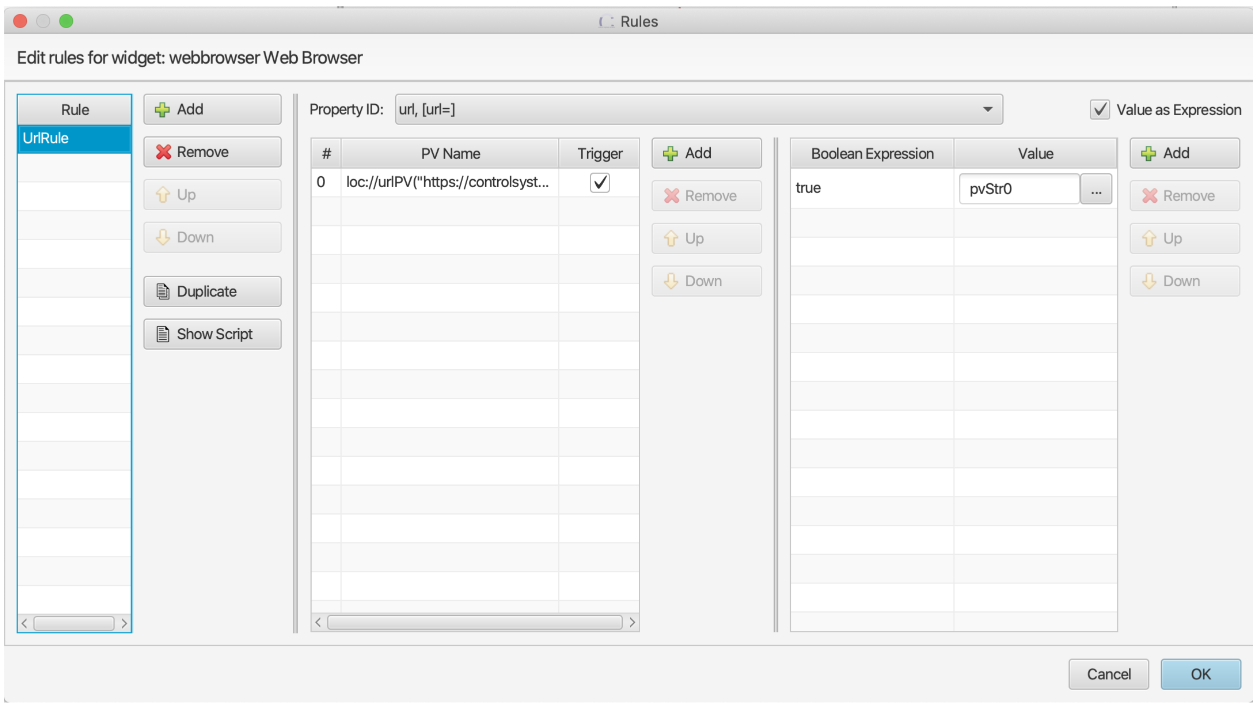
### Boolean Expression

The “Boolean Expression” is JavaScript boolean expression, so all JavaScript operators are applicable here. All input PVs of the rule are accessible in the expression.

- Numeric double values are referenced using the syntax pv{index}, e.g. pv0 > pv1.
- Numeric long and integer values are referenced using the syntax pvInt{index}, e.g. pvInt0 > pvInt1.
- String values are referenced using the syntax pvStr{index}, e.g. pvStr0 == “apple”.
- A PV severity value is referenced using the syntax pvSev{index}, e.g. pvSev0 == 1. Severity values are:
  - 0 - OK
  - 1 - Minor
  - 2 - Major
  - 3 - Invalid
  - 4 - Undefined

### Value as Expression

In addition to outputting a constant value to the property based on the boolean expression value, you may also output an expression value to the property. For example, if you want use the string value of a PV as the URL of a web browser widget, you can set the output expression to pvStr0. If you want to skip the boolean expression, simply set boolean expression to `true`.



Edit rules for widget: webbrowser Web Browser

| Rule |
|---|
| UrlRule |

Property ID: url, [url=]

Value as Expression ✓

| # | PV Name | Trigger |
|---|---|---|
| 0 | loc://urlPV("https://controlsyst... | ✓ |

| Boolean Expression | Value |
|---|---|
| true | pvStr0 |

## Scripts

**DISCLAIMER**

Scripting in CS Studio should be used **only** if a particular task cannot be achieved with the built-in features of the various widgets, or with formula functions or rules. In particular, CS Studio **should not** be used as a script execution platform for data processing or complex business logic. It should also be noted that each script referenced by the widgets of an OPI is compiled when the OPI is loaded, which adds to the overall load time, and which may degrade the initial responsiveness of the UI.

Supported script language versions are Python 2.7 and JavaScript 1.7 (ECMAScript support is **very** limited).

For complex dynamic behaviors which cannot be achieved by formula functions or rules, you can attach one or more JavaScript or Python scripts to a widget or display. Both script types accept PVs as inputs. Script execution is triggered by the value change event of input trigger PVs, i.e. a change of PV value or timestamp. In a script the value, timestamp or severity of the input PVs are accessible, see access_pv_in_script. The widget and display objects are also accessible in script, see access_widget .

Both JavaScript and Python script may call Java code by importing corresponding packages. For example:

**JavaScript Example:**

```
importPackage(Packages.org.eclipse.jface.dialogs);
MessageDialog.openInformation(null, "Dialog from JavaScript", "This is a dialog␣
↪opened from JavaScript")
```

**Python script Example:**

```
from org.eclipse.jface.dialogs import MessageDialog
MessageDialog.openInformation(None, "Dialog from Python", "This is a dialog opened␣
↪from Python")
```

As seen above, calling Java code is very similar between JavaScript and Python. Most script examples in this help document are in JavaScript, but it should be easy for you to translate them to Python. For example, here are two code snippets written in JavaScript and Python respectively. The functionality in the scripts is identical.

**JavaScript Example:**

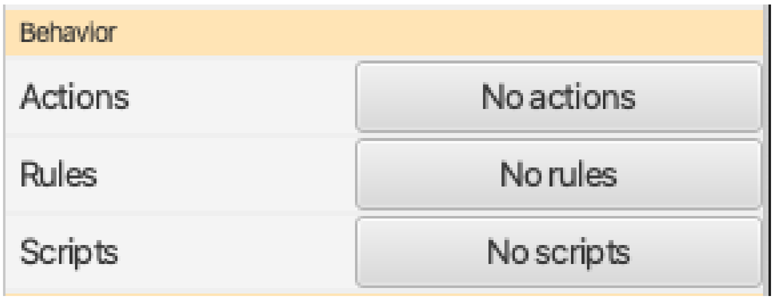
```
importPackage(Packages.org.csstudio.opibuilder.scriptUtil);
var value = PVUtil.getDouble(pvs[0]);
var RED = ColorFontUtil.RED;
widget.setPropertyValue("start_angle", value);
widget.setPropertyValue("foreground_color", RED);
```

**Python script Example:**

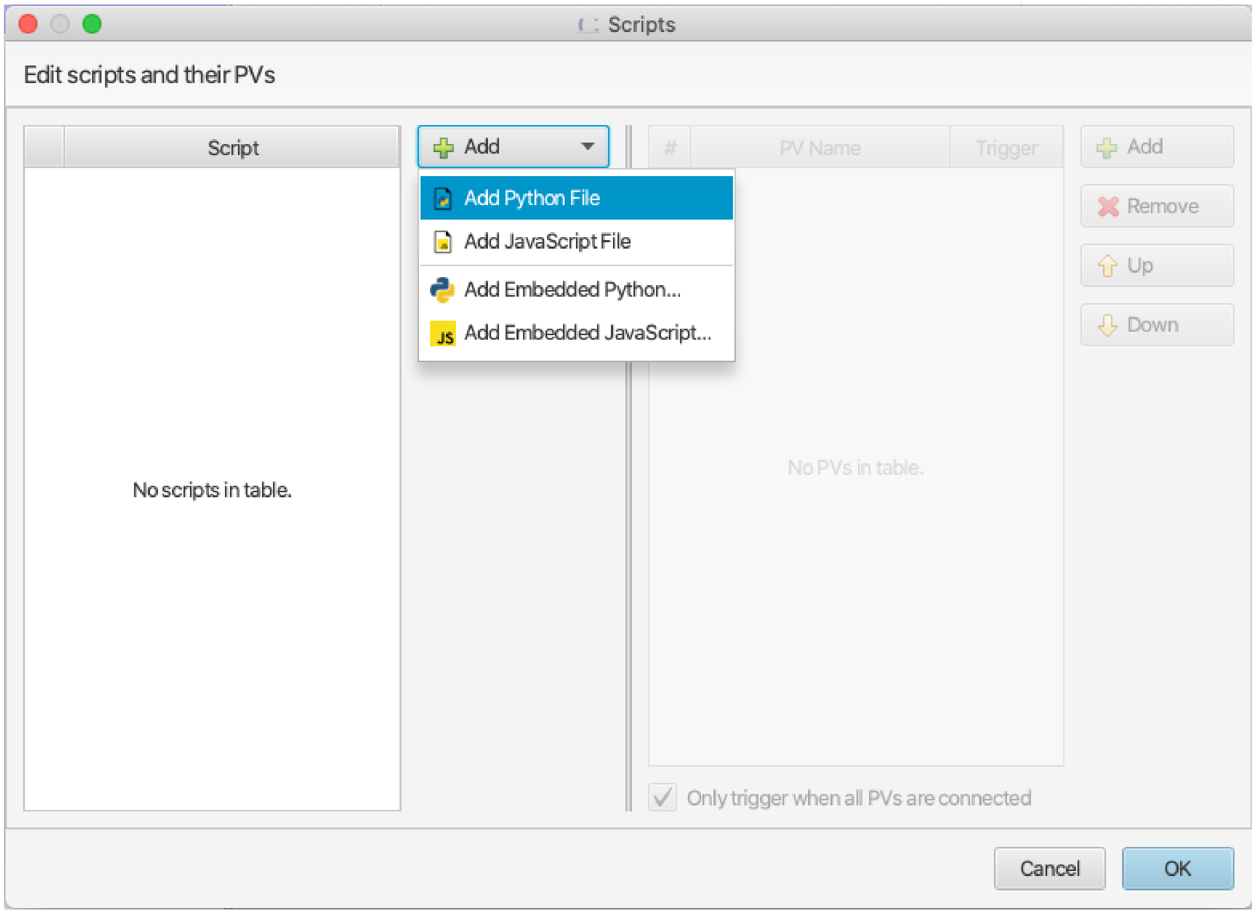
```
from org.csstudio.opibuilder.scriptUtil import PVUtil
from org.csstudio.opibuilder.scriptUtil import ColorFontUtil
value = PVUtil.getDouble(pvs[0])
RED = ColorFontUtil.RED
widget.setPropertyValue("start_angle", value)
widget.setPropertyValue("foreground_color", RED)
```

**Steps to attach scripts to a widget:**

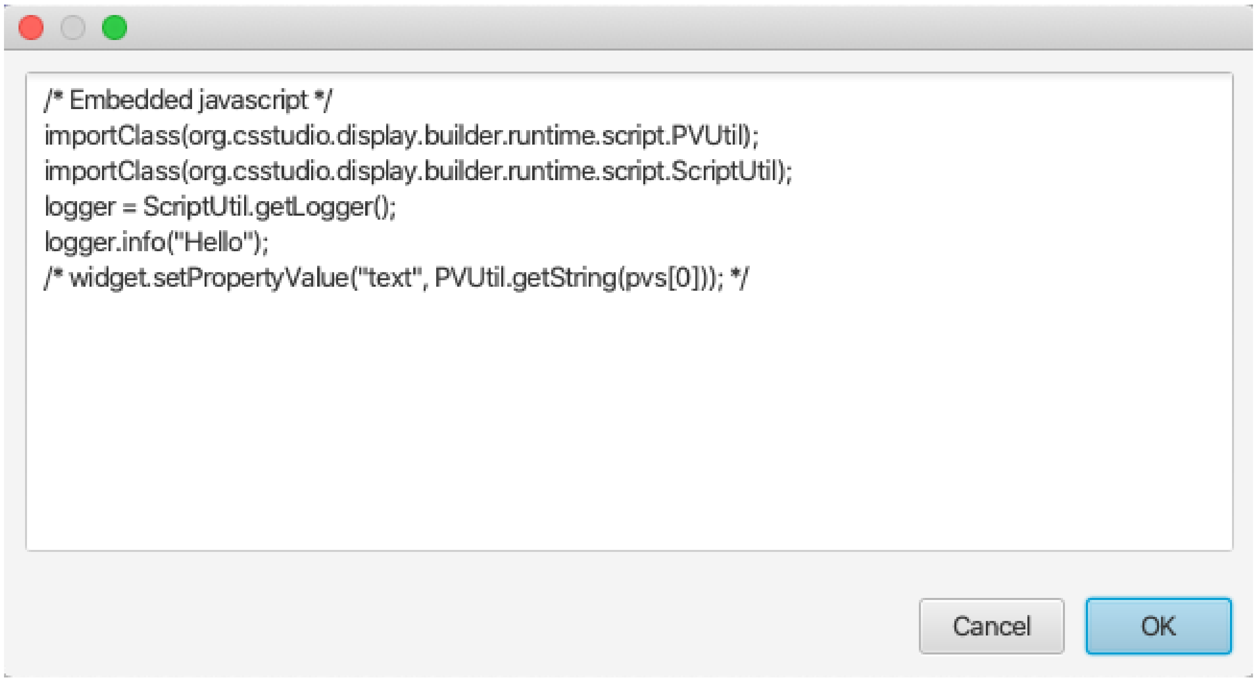
1. In the widget properties pane, click the button to launch the script editor dialog:



2. In script editor dialog you can add one or multiple scripts. A script may be either a script file on the file system, or a code block embedded into the OPI file.

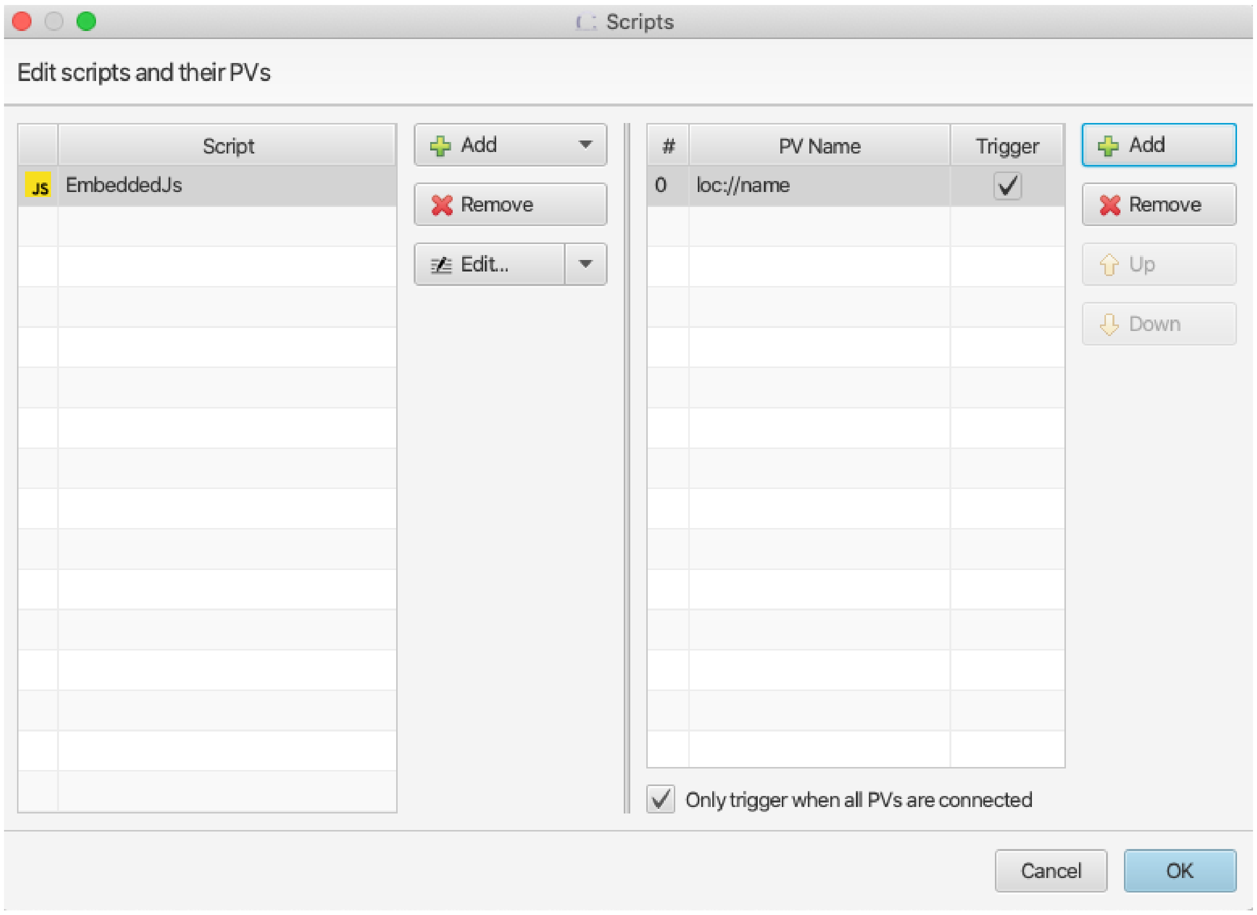


3. If an embedded script is selected, the editor will create a small script template and show it in an editor window.



```
/* Embedded javascript */
importClass(org.csstudio.display.builder.runtime.script.PVUtil);
importClass(org.csstudio.display.builder.runtime.script.ScriptUtil);
logger = ScriptUtil.getLogger();
logger.info("Hello");
/* widget.setPropertyValue("text", PVUtil.getString(pvs[0])); */
```

4. For each script you can specify one or multiple input PVs. PVs that trigger script execution should be checked in Trigger column. There must be at least one trigger PV for each script.



| | Script |
|---|---|
| JS | EmbeddedJs |

| # | PV Name | Trigger |
|---|---|---|
| 0 | loc://name | ✓ |

✓ Only trigger when all PVs are connected

### Internals

## 8.19 3D Viewer

### 8.19.1 Overview

The 3d Viewer is a tool that allows users to configure 3 dimensional structures using spheres, cylinders, and boxes.

These structures are defined in shape file (**.shp*) and parsed by the application.

The resultant structure is then rendered on screen. This structure can be viewed in the application which allows rotation, zoom, and movement.

The individual spheres, cylinders, and boxes that make up the structure can have their coordinates, sizes, and colors specified.

### 8.19.2 Shape (.shp) File Syntax

The viewer parses shape files from beginning to end. Any error in the shape file will cause the parsing of the entire file to fail and no resulting structure will be rendered.

**Comments** Shape files can have comments. A comment is a line of text that starts with a '#'. This line will be ignored when parsing the shape file.

**Background Color** The background color of the viewer can be controlled using the following command.

```
background(r, g, b, A)
```

This command has four parameters. The red, green, and blue values for the color are the first three. Each color value is an integer from 0 through 255. These are followed by the alpha value which allows you to control the transparency of the color. Alpha is a floating point number in the range [0, 1]. An alpha of 0 is transparent while an alpha of 1 is opaque.

Multiple background colors may be defined, however only the last defined background color will be used.

**Spheres** A sphere may be defined using the following command.

```
sphere(x, y, z, R, r, g, b, A)
```

This command has eight parameters. The first three parameters are the x, y, and z values which represent the center point of the sphere in the three dimensional space. The x, y, and z parameters are floating point values. These are followed by the radius of the sphere, another floating point value. The final four parameters are the red, green, blue, and alpha values used to define the color of the sphere. Red, green, and blue are integer values in [0, 255] and alpha is a floating point value in [0, 1].

**Cylinders** A cylinder may be defined using the following command.

```
cylinder(x1, y1, z1, x2, y2, z2, R, r, g, b, A)
```

This command has eleven parameters. The first three parameters are the x, y, and z values which represent one end point of the cylinder. The second three parameters are the x, y, and z values which represent the other end point of the cylinder. All x, y, and z parameters are floating point values. These are followed the cylinder's radius. The radius is a floating point value. The final four parameters are the red, green, blue, and alpha values used to define the color of the sphere. Red, green, and blue are integer values in [0, 255] and alpha is a floating point value in [0, 1].

**Boxes** A box may be defined using the following command.

```
box(x1, y1, z1, x2, y2, z2, r, g, b, A)
```

This command has 10 parameters. The first three parameters are the x, y, and z values which represent one corner of the box.The second three parameters are the x, y, and z values which represent the opposite corner of the box. All x, y, and z parameters are floating point values. The final four parameters are the red, green, blue, and alpha values used to define the color of the sphere. Red, green, and blue are integer values in [0, 255] and alpha is a floating point value in [0, 1].

If the first corner of a box was defined at (0, 0, 0) and the second corner at (100, 100, 100) then the box would have one corner at the origin and one corner at (100, 100, 100). Each of the boxes sides would be of length 100, and the boxes center point would be (50, 50, 50).

**Cones** A cone may be defined using the following command.

```
cone(x1, y1, z1, R, x2, y2, z2, r, g, b, A)
```

This command has eleven parameters. The first three parameters are the x, y, and z values of the base, followed by the radius of the base. The second three parameters are the x, y, and z values of the tip of the cone. The final four parameters are the red, green, blue, and alpha values used to define the color.

**Tool Tips** A final string added to a shape defines a tool tip for the shape.

### 8.19.3 Example Shape File

```
# This is a comment. It will be ignored when the file is parsed.

# The background of the viewer is set to be nearly black.
```

(continues on next page)

(continued from previous page)

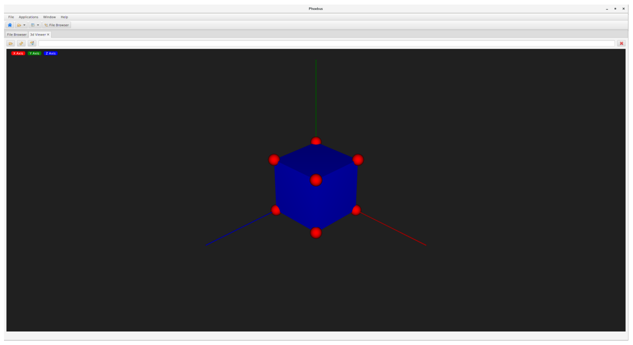
```
background(32, 32, 32, 1)

# A red sphere of radius 10, is placed at each corner of the box we are about to␣
↪define.
sphere(  0,   0,   0, 10, 255, 0, 0, 1, "Origin")
sphere(100,   0,   0, 10, 255, 0, 0, 1)
sphere(  0,   0, 100, 10, 255, 0, 0, 1)
sphere(100,   0, 100, 10, 255, 0, 0, 1)
sphere(  0, 100,   0, 10, 255, 0, 0, 1)
sphere(100, 100,   0, 10, 255, 0, 0, 1)
sphere(  0, 100, 100, 10, 255, 0, 0, 1)
sphere(100, 100, 100, 10, 255, 0, 0, 1)

# A blue box is defined with one corner at the origin, and the opposite
# corner at (100, 100, 100). This will result in a cube with each side
# being of magnitude 100.
box(0, 0, 0, 100, 100, 100, 0, 0, 255, 1)

# Cone along the X axis, base at x=200, radius 20, tip at x=300
cone ( 200,   0,   0, 10,   300,   0,   0,   255, 100, 100, 1, "X")
```

**Resulting Structure**



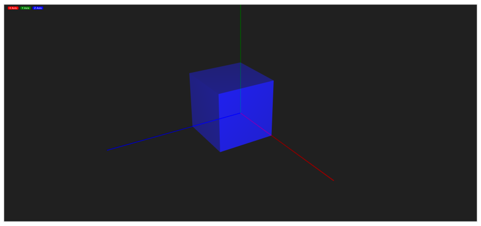
## 8.19.4 Transparency

JavaFX does not sort 3D objects by depth. What this means is that you have to be thoughtful of the order you add 3D shapes to a scene. For example, if a sphere needed to be displayed inside a translucent box, the sphere would have to be added *before* the box. If the box first were added first, it would still be translucent, but the JavaFX renderer would not draw the sphere because it doesn't sort the scene graph by depth.

**Examples**

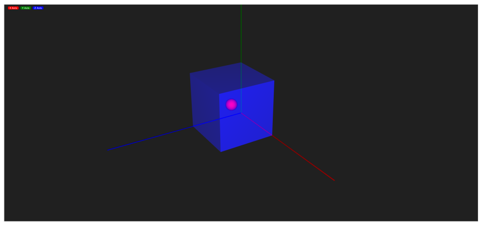
**Here, the box is added first and the sphere is not drawn.**

```
background(32, 32, 32, 1)
box(0, 0, 0, 100, 100, 100, 0, 0, 255, 0.1)
sphere(50, 50, 50, 10, 255, 0, 0, 1)
```



**Here, the box is added second and the sphere is drawn correctly.**

```
background(32, 32, 32, 1)
sphere(50, 50, 50, 10, 255, 0, 0, 1)
box(0, 0, 0, 100, 100, 100, 0, 0, 255, 0.1)
```



### 8.19.5 Compatibility

The 3D Viewer requires support from the graphics system.

Known to work:

- OpenJDK 11 on Mac OS 10.13.6
- OpenJDK 11 on Windows 10
- Oracle JDK 9, Oracle JDK 10, and OpenJDK 11 on RHEL 7.6

Running on Linux requires direct graphics on a local machine.

Can be made to work:

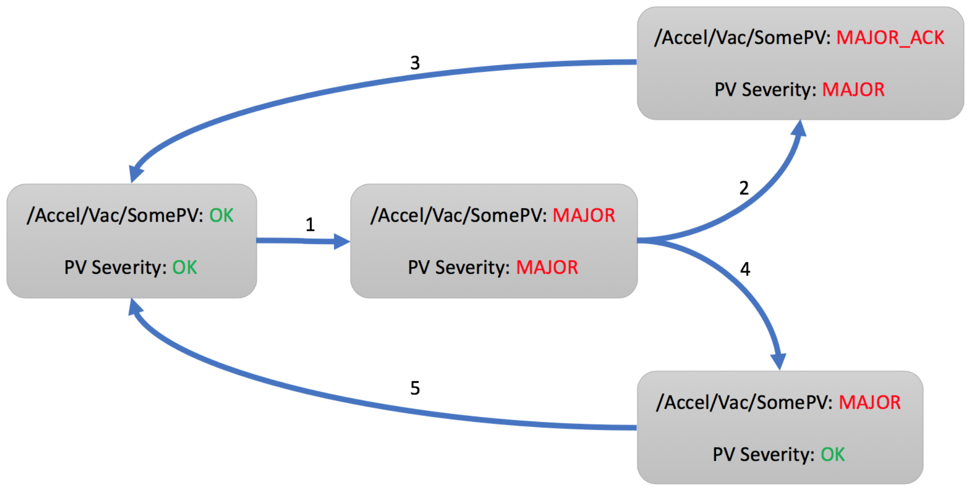
- Oracle JDK 10 or OpenJDK 11 on Centros 7.5 running inside VirtualBox, hosted on RHEL 7.6, when setting `-Dprism.forceGPU=true`

In case of problems which usually include error messages `System can't support ConditionalFeature.SCENE3D`, start the program with `-Dprism.verbose=true` and `-Djdk.gtk.verbose=true`

## 8.20 Alarms

### 8.20.1 Overview

The alarm system monitors the alarm state for a configurable list of PVs. When the alarm severity of any PV changes from *OK* to for example *MAJOR*, the alarm system changes to that same alarm severity (transition 1 in the diagram below).



For the overall alarm to return to *OK*, two things need to happen:

- The alarm severity of the PV must return to *OK*
- The alarm must be acknowledged

Typically, the alarm will persist for a while. A user acknowledges the alarm (2) and starts to address the underlying issue. Eventually, the reason for the alarm is removed, the severity of the PV recovers to *OK*, and the alarm system returns to an overall *OK* state (3).

It is also possible that the underlying issue is short lived, and the PV recovers to *OK* on its own. The alarm system latches the alarm, so users can see that there was an alarm (4). When the user acknowledges the alarm, the system returns to an overall *OK* state (5).

The order of PV recovery and acknowledgement does therefore not matter. There are two more details which are not shown in the diagram.

The alarm system maximizes the alarm severity of a PV. Assume a PV enters the alarm state (1) because its severity is *MINOR*. The alarm state will also be *MINOR*. If the PV severity now changes to *MAJOR*, the alarm state will become *MAJOR* as well. Should the PV severity now return to *MINOR*, the alarm state will remain *MAJOR* because the alarm system takes note of the highest PV severity. As already shown in (4), a PV severity clearing to *OK* still leaves the alarm state at the highest observed severity until acknowledged.

Finally, while alarms will by default *latch* as described above, an alarm can be configured to not latch. When such a non-latching PV enters an alarm state (1), once the PV recovers, it will right away return to *OK* via (4) and (5) without requiring acknowledgement by an end user.

Note that the alarm system reacts to PVs. Details of how PVs generate alarms, for example at which threshold an analog reading would enter a *MINOR* alarm state are determined in the control system. The alarm system can notify users of an alarm, but it cannot explain why the alarm happened and what the user should do. Each alarm should be configured with at least one “guidance” message to explain the alarm and a “display” link to a related control system screen.

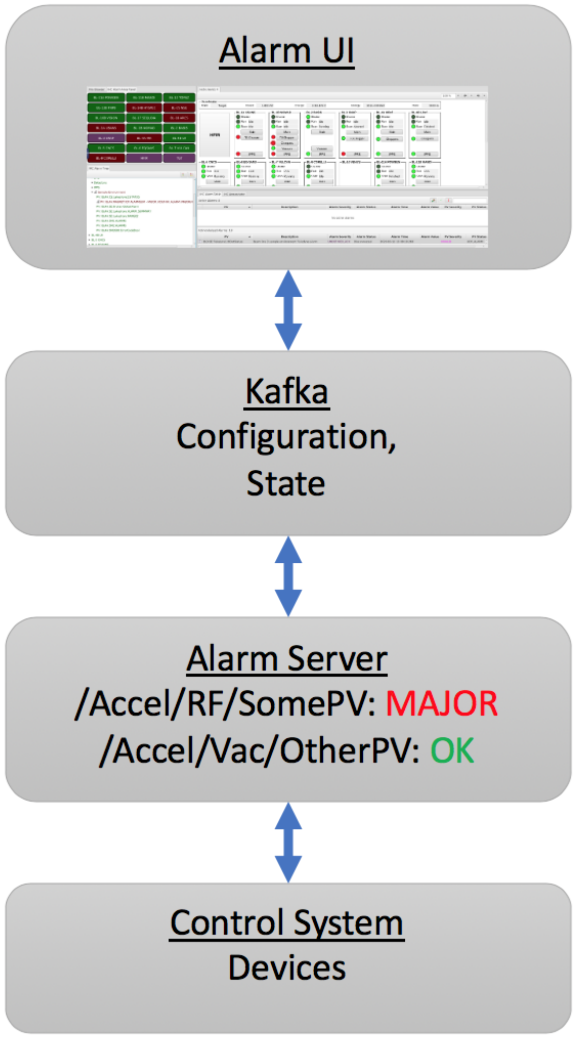
### 8.20.2 Components

The alarm system consists of an alarm server and a user interface.

The Alarm Server monitors a set of PVs, tracking their alarm state. The alarm server tracks updates to the PVs received from the control system.

The user interface shows the current alarms, allows acknowledgement, and provides guidance, links to related displays.

Kafka stores the alarm system configuration, and provides the communication bus between the alarm server and user interface.



Refer to *applications/alarm/Readme.md* for setting up Kafka and the alarm server.

### 8.20.3 User Interface

The UI includes the following applications:

- Alarm Tree: Primarily used to configure the alarm system, i.e. to add PVs and define their alarm details.

  The alarm configuration is hierarchical, starting from for example a top-level *Accelerator* configuration to components like *Vacuum*, *RF*, with alarm trigger PVs listed below those components. Configuration settings for *Guidance*, *Displays* etc. are inherited along the hierarchy, so that all alarm under */Accelerator/Vacuum* will see all the guidance and displays configured on *Vacuum*.

  The alarm system does not enforce how the hierarchical configuration is used. The 'components' could be subsystems like *Vacuum*, *RF*, or they could refer to areas of the machine like *Front End*, *Ring*, *Beam Line*. There can be several levels of sub-components, and each site can decide how to arrange their alarm trigger PVs to best re-use guidance and display information so that the configuration of individual PVs is simplified by benefitting from the inherited settings along the hierarchy.

- Alarm Table: Main runtime interface, shows current alarms.

  Ideally, this table will be empty as the machine is running without issues. Once alarms occur, they are listed in a table that users can sort by PV name, description, alarm time etc.

  The context menu of selected alarms offers links to guidance messages and related displays.

  Alarms can be acknowledged, which moves them to a separate table of acknowledged alarms.

- Alarm Area Panel: Shows summary of top-level alarm hierarchy components.

  Useful as a basic alarm status indicator that can be checked "across the room".

- Annunciator: Annunciates alarms.

  Optional component for voice annunciation of new alarms.

Each of the above alarm apps can be launched from the menu. They can also be opened from the command line as follows:

```
-resource 'alarm://localhost/Accelerator?app=alarm_tree'
-resource 'alarm://localhost/Accelerator?app=alarm_table'
-resource 'alarm://localhost/Accelerator?app=alarm_area'
```

### 8.20.4 Alarm Configuration Options

Alarm configurations are imported into the Alarm Server in an XML format, the schema for which may be found here. The options for an entry in the hierarchical alarm configuration always include guidance, display links etc. as described further below. In addition, alarm PV entries have the following settings.

#### Description

This text is displayed in the alarm table when the alarm triggers.

The description is also used by the alarm annunciator. By default, the annunciator will start the actual message with the alarm severity. For example, a description of "Vacuum Problem" will be annunciated as for example "Minor Alarm: Vacuum Problem". The addition of the alarm severity can be disabled by starting the description with a "*" as in "* Vacuum Problem".

When there is a flurry of alarms, the annunciator will summarize them to "There are 10 more alarms". To assert that certain alarms are always annunciated, even if they occur within a burst of other alarms, start the message with "!" (or "*!").

#### Behavior

- Enabled: De-select to disable an alarm, i.e. to ignore the value of this alarm trigger PV.
- Latch: By default, alarms latch to the highest received severity until the alarm is acknowledged and clears. De-select if the alarm should recover without requiring acknowledgement.
- Annunciate: Should the alarm be annunciated (if the annunciator is running), or should it only be displayed silently?
- Alarm Delay: Only alarm if the trigger PV remains in alarm for at least this time, see examples below.

- Alarm Count: Used in combination with the alarm delay. If the trigger PVs exhibits a not-OK alarm severity more than 'count' times within the alarm delay, recognize the alarm.

  For example, an alarm delay of 10 with an alarm count of 5 means: Recognize an alarm if the PV enters a not-OK severity for more than 10 seconds, or more often than 5 times within 10 seconds.

  When the count is zero, only the alarm delay is used.

- Enabling Filter: An optional expression that can enable the alarm based on other PVs.

  Example: *'abc' > 10* will only enable this alarm if the PV 'abc' has a value above 10.

The Alarm Delay and Count work in combination. By default, with both the alarm delay and count at zero, a non-OK PV severity is right away recognized. When the alarm delay is larger than zero, it starts a timer to check the PV after the given delay. For example, assume an alarm delay of 10 seconds, and the PV enters a MINOR alarm. If the PV still carries a not-OK severity after 10 seconds, the alarm state becomes MINOR or whatever the highest alarm severity of the PV was in the 10 seconds since first entering a not-OK severity. On the other hand, if the PV recovers to OK, there will be no alarm after the 10 second delay.

As a second example, consider a PV that assumes MINOR severity, then recovers to OK and re-enters MINOR severity a couple of times. If the non-OK severity never persists longer then 10 seconds, it is ignored. The alarm count can be used to detect such cases. With an alarm count of 5, even if each non-OK severity lasts only say 1 second, when the PV becomes not-OK for 5 or more times within 10 seconds, the alarm will be indicated. For a delay of 10 seconds and a count of 5, there are thus two ways to enter an alarm state: Either the PV stays not-OK for at least 10 seconds, or it briefly becomes not-OK for at least 5 times within 10 seconds.

While the filter, alarm delay and count can be helpful to reduce the number of alarms from 'noisy' PVs, ideally all such logic is implemented at the source, i.e. in the IOC that provides the alarm trigger PV. This not only simplifies the task of the alarm system, but also makes the behavior more obvious, since a PV is used "as is", the alarm server uses the same alarm state that is indicated in a display panel, without adding filtering that might not be obvious when later inspecting an alarm.

Note again that the alarm system only reacts to the severity of alarm trigger PVs. For EPICS records, this is for example configured via the HIGH, HSV and HYST fields of analog records, or the ZSV and OSV fields of binary records. Why, when and for how long an alarm trigger PV enters an alarm state is configured on the data source, and is not immediately obvious from the received alarm severity.

For example, an analog record might enter a MINOR alarm state when its value exceeds the 'HIGH' value. Why a certain HIGH threshold was chosen, what the user should do about it, and how the threshold could be changed, however, cannot be automatically determined. When adding an alarm trigger PV to the alarm system, it is therefore important to also configure guidance and display links which allow the user to figure out:

- What does this alarm mean? What should I do about it?
- What displays allow me to see more, where can I do something about the alarm?

### 8.20.5 Guidance

Each alarm should have at least one guidance message to explain the meaning of an alarm to the user, to list for example contact information for subsystem experts. Guidance can be configured on each alarm PV, but it can also be configured on parent components of the alarm hierarchy.

- Title: A short title for the guidance that will appear in the context menu of the alarm, for example "Contacts" or "What to do".
- Detail: A slightly longer text with the content of the guidance, for example a list of telephone numbers, or description of things to try for handling the alarm.

### 8.20.6 Displays

As with Guidance, each alarm should have at least one link to a control system display that shows the actual alarm PV and the surrounding subsystem.

- Title: Short title for the display link that will appear in the context menu, for example "Vacuum Display".
- Detail: The display link. This is handled similar to *-resource..* arguments passed on the command line. For plain display files, the complete path to the file will suffice, and the display tool is recognized by the file extension, i.e. **.bob* for the display runtime, or **.html* to open a web page. When passing macros, a complete URL is required.

Examples:

```
/path/to/display.bob
http://server.site/path/to/display.bob
http://server.site/path/to/display.bob?MACRO=Value&ANSWER=42
file:///path/to/display.bob?MACRO=Value&OTHER=42$NAME=Text+with+spaces
```

### 8.20.7 Automated Actions

Automated actions are performed when the node in the alarm hierarchy enters and remains in an active alarm state for some time.

The intended use case for automated action is to for example send emails in case operators are currently unable to acknowledge and handle the alarm. If the alarm should always right away perform some action, then this is best handled in the IOC.

The automated action configuration has three parts:

- Title: The "Title" can be set to a short description of the action.
- Delay: The "Delay", in seconds, determines how long the node needs to be in an active alarm state before the automated action is executed. A delay of 0 seconds will immediately execute the action, which in practice suggests that the action should be implemented on an IOC.
- Detail: The "Detail" determines what the automated action will do.

`mailto:user@site.org,another@else.com`: Sends email with alarm detail to list of recipients.

The email server is configured in the alarm preferences.

`cmd:some_command arg1 arg2`: Invokes command with list of space-separated arguments. The special argument "*" will be replaced with a list of alarm PVs and their alarm severity. The command is executed in the `command_directory` provided in the alarm preferences.

`sevrpv:SomePV`: Names a PV that will be updated with the severity of the alarm, i.e. a value from 0 to 9 to represent the acknowledged or active alarm state. The delay is ignored for `sevrpv:` actions.

Suggested PV template:

```
# Example for "Severity PV"
# used with automated action set to "sevrpv:NameOfPV"
#
# softIoc -s -m N=NameOfPV -d sevrpv.db

record(mbbi, "$(N)")
{
    field(ZRVL, 0)
    field(ZRST, "OK")
```

(continues on next page)

(continued from previous page)

```
    field(ONVL, 1)
    field(ONST, "MINOR_ACK")
    field(ONSV, "MINOR")
    field(TWVL, 2)
    field(TWST, "MAJOR_ACK")
    field(TWSV, "MAJOR")
    field(THVL, 3)
    field(THST, "INVALID_ACK")
    field(THSV, "INVALID")
    field(FRVL, 4)
    field(FRST, "UNDEFINED_ACK")
    field(FRSV, "INVALID")
    field(FVVL, 5)
    field(FVST, "MINOR")
    field(FVSV, "MINOR")
    field(SXVL, 6)
    field(SXST, "MAJOR")
    field(SXSV, "MAJOR")
    field(SVVL, 7)
    field(SVST, "INVALID")
    field(SVSV, "INVALID")
    field(EIVL, 8)
    field(EIST, "UNDEFINED")
    field(EISV, "INVALID")
    field(INP,  "0")
    field(PINI, "YES")
}
```

### Inclusions

The Phoebus alarm server supports Xinclude, allowing for the breakup of hierarchies into multiple files.

```
<?xml version='1.0' encoding='utf8'?>
<config name="HeartOfGold">
    <pv name="NUTRIMATIC">
        <enabled>true</enabled>
        <latching>false</latching>
        <annunciating>false</annunciating>
        <description>Does not make tea</description>
        <delay>10</delay>
        <count>30</count>
    </pv>
    <pv name="INFINITE:IMPROBABILITY:DRIVE">
        <enabled>true</enabled>
        <latching>false</latching>
        <annunciating>false</annunciating>
        <filter> Marvin + Eddie == 0</filter>
    </pv>
    <xi:include href="/path/to/inclusion/file/includion_file.xml" xpointer="include-
↪component" xmlns:xi="http://www.w3.org/2001/XInclude"/>
</config>
```

Where the include component is identified in the inclusion file with a DID declared id component:

```xml
<?xml version='1.0' encoding='utf8'?>
<!DOCTYPE config [
  <!ATTLIST component id ID #IMPLIED>
]>
<config name="GPP">
    <component name="GPP" id ="component">
        <pv name="EDDIE">
            <enabled>true</enabled>
            <latching>false</latching>
            <annunciating>false</annunciating>
            <description>Eddie the Computer</description>
        </pv>
        <pv name="MARVIN">
            <enabled>true</enabled>
            <latching>true</latching>
            <annunciating>false</annunciating>
            <description>Paranoid android</description>
            <delay>100</delay>
            <count>1000</count>
        </pv>
    </component>
</config>
```

## 8.21 Alarms Logging

### 8.21.1 Overview

In order for the alarm system to be effective and functional, proper configuration and periodic tuning is essential. While the Alarm Server monitors a set of PVs, tracking their alarm state, the alarm logging service archives all the state, configuration and commands associated with the Alarm Server.

The alarm logging table provides a user interface to the alarm logging service allowing users to effectively search, filter, and parse through the archived messages to better configure the Alarm Server.

## 8.22 Alarm Datasource

### 8.22.1 Overview

The Alarm System allow users to view and handle the alarms triggered by Process Variables (PVs) on front-end computers (IOCs). There are a set of dedicated applications (Alarm tree, table, and panel) which display alarms and alarm related information (like guidance). These applications also provide the ability to take a set of actions needed to effectively handle an alarm.

The alarm datasource provides a subsection of the alarm information and functionality. This makes it possible for user to access beast alarm information of any other cs-studio application. OPI screens can now embed informatino about from the alarm server, like the acknowledgement state of a pv, etc..

### 8.22.2 PV syntax

The standard prefix for the datasource is `alarm://` which can be omitted if configured as the default datasource.

`alarm` pvs can be used to connect to any node or leaf from the alarm tree. The alarm tree represents the configured hierarchy of the alarm pv's, the hierarchy consists of multiple nodes and alarming pv's

Node(Area): Front end, DTL, Linac, Storage Ring, . . .
Node(System): ICS, Cooling, Vacuum, RF, . . .
Node(Optionally sub-system): ICS/IOCs, RF/LLRF, RF/HPRF, . . .
Alarm Trigger PVs on the final level.

**You can create a alarm channel for any Alarm Trigger PVs or for any Area, System, Subsystem.**

```
# alarm://complete_path_to_area
alarm://NSLS2_OPR/Storage ring/Diagnostics

# alarm://complete_path_to_system
alarm://NSLS2_OPR/Linac/Controls/Timing/LN-TS{EVR}Alrm:Link-Sts
```

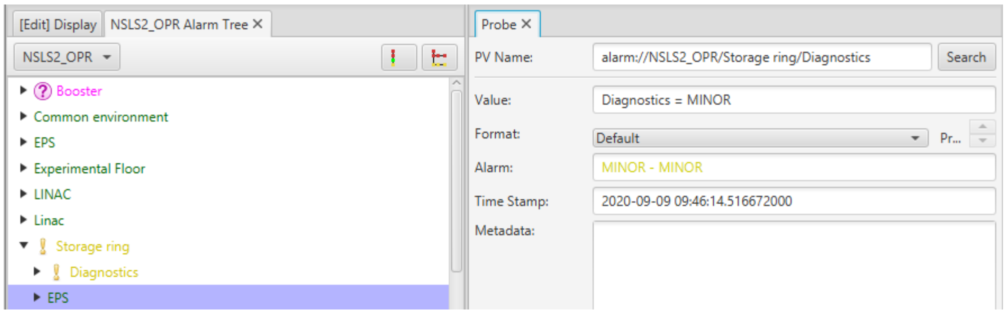
### Reading alarm pv's

The `alarm://` pvs return a VString describing the alarm state of the associated node or leaf in the alarm tree

e.g.

Connecting to a node
`alarm://NSLS2_OPR/Storage ring/Diagnostics`
returns
"Diagnostics = MINOR"

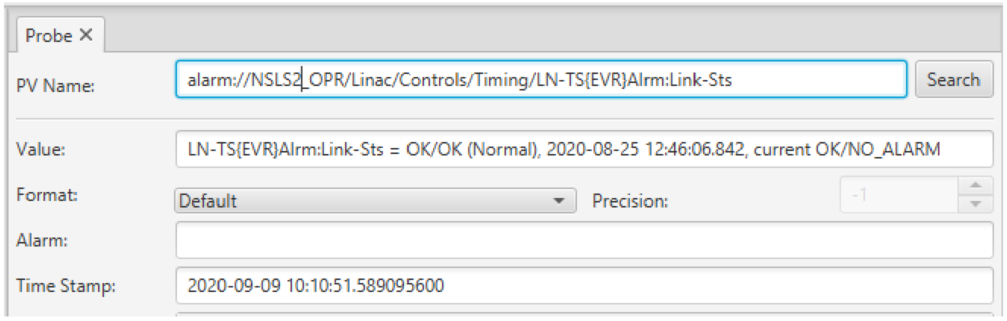
The return value is a VString, with the string value describing the alarm state of the node. The Alarm meta data of the pv also matches the alarm state.



PV Name: alarm://NSLS2_OPR/Storage ring/Diagnostics
Value: Diagnostics = MINOR
Format: Default
Alarm: MINOR - MINOR
Time Stamp: 2020-09-09 09:46:14.516672000
Metadata:

Connecting to a leaf
`alarm://NSLS2_OPR/Linac/Controls/Timing/LN-TS{EVR}Alrm:Link-Sts`
returns
"LN-TS{EVR}Alrm:Link-Sts = OK/OK (Normal), 2020-08-25 12:46:06.842, current OK/NO_ALARM"

The return value is a VString, with the string value describing the alarm state along with a description of the pv's alarm value if present and its current alarm value. The Alarm meta data of the pv also matches the alarm state.


Probe

PV Name: alarm://NSLS2_OPR/Linac/Controls/Timing/LN-TS{EVR}Alrm:Link-Sts

Search

Value: LN-TS{EVR}Alrm:Link-Sts = OK/OK (Normal), 2020-08-25 12:46:06.842, current OK/NO_ALARM

Format: Default

Precision: -1

Alarm:

Time Stamp: 2020-09-09 10:10:51.589095600

### Special Fields

The alarm pvs have a few additional fields which can be used to access specific attributes of the alarm node/leaf

These fields can be used by appending the following field tag at the end the alarm pv.

e.g.

```
alarm://NSLS2_OPR/Linac/Controls.state
```

**.state**
Returns an Enum indicating the current alarm state of the alarm node/leaf.

**.active**
Return a Boolean true if the alarm pv is active. An active alarm pv is one which is currently in an alarm state which AND it has not been acknowledged.

**.enabled**
Returns a Boolean true if the alarm pv is enabled. This is a writeable field which can be used to enabled or disabled the associated alarm element in the tree.

**.duration**
Returns a String with the duration since this alarm pv has been in an alarm state.

### Writing to alarm pv's

```
alarm://NSLS2_OPR/SR/Vacuum
```

### acknowledge

The alarm pvs can be used to acknowledge or unacknowledge parts of the alarm tree. The alarm pvs accept String, boolean, and Number values which are interpreted as follows:

When writing **strings**

"ack" or "acknowledge"

Acknowledge all alarms associated with the node or leaf of the alarm tree associated with this alarm pv

"unack" or "unacknowledge"

Unacknowledge all the alarms associated with the node or leaf of the alarm tree associated with this alarm pv

When writing **Numbers**

Writing any non zero number is treated as an acknowledge while 0 will unacknowledge an alarm

When writing **booleans**

A "true" value is to acknowledge an alarm while a "false" value will unacknowledge an alarm

#### enable

The alarm pvs can be used to enable or disable parts of the alarm tree. The alarm pvs accept String values which are interpreted as follows:

When writing **strings**

"enable"
Enable all alarms associated with the node or leaf of the alarm tree associated with this alarm pv

"disable"
Disable all the alarms associated with the node or leaf of the alarm tree associated with this alarm pv

## 8.23 Logging Configuration

Allows for the runtime configuration of loggers.

Application developers are encouraged to use `java.util.logging` for log messages.

To debug a problem, the logging configuration allows you to for example change the log level for the `org.phoebus.pv` module to use FINE or even ALL to see all received PV value updates.

## 8.24 Update

The 'update' application allows a product to self-update.

Assume you have a site-specific product, i.e. a ZIP file that users at your site can download. By including the 'update' application in your product and setting two preference settings, your product can self-update.

### 8.24.1 Configuration

The `current_version` setting needs to provide the current version of your product in the format `YYYY-MM-DD HH:MM`. The `update_url` setting needs to point to a file or web URL that contains the latest product. You most likely need to create separate products for each architecture, because for example the JavaFX libraries are specific to an architecture, and you want system-specific launch scripts, batch files or Mac apps. The URL can thus contain `$(arch)` which will be will be replaced by "linux", "mac" or "win".

Example:

> org.phoebus.applications.update/current_version=2018-06-18 13:10 org.phoebus.applications.update/update_url=http://my.site.org
> for-my-site-$(arch).zip

There are additional settings that allow re-writing the file names of the downloaded ZIP file or skipping files. For details, see full description of the update preference settings.

### 8.24.2 Usage

On startup, the update mechanism checks the `update_url`. If that file is dated after the `current_version`, an "Update" button is added to the status bar to indicate that an update is available.

Clicking that "Update" button opens a dialog with details on the current version, the updated version, and the installation location that will be replaced in the update.

When pressing "OK", the update is downloaded into an `update/` folder below your current install location. Finally, a prompt indicates completion of the update, and the product exists for you to start the updated version. Your launch script needs to check for the presence of an `update/` folder. If one is found, it can replace the current installation with the content of the update folder, delete it, and start the new version.

### 8.24.3 Details

Earlier versions directly replaced the `lib/*.jar` files with a downloaded update, but on Windows these are locked by the running instance. The downloaded files are thus staged in an `update/` folder, and the launcher script replaces the previous files with the new ones before starting the JVM which then locks the files as they are used.

## 8.25 PV Table

The PV Table provides a tabular view of PV names and their current value with time stamp and alarm state.

You can take a "snapshotData" of current values and dates, and the table will now highlight rows where the current value differs from the snapshotData.

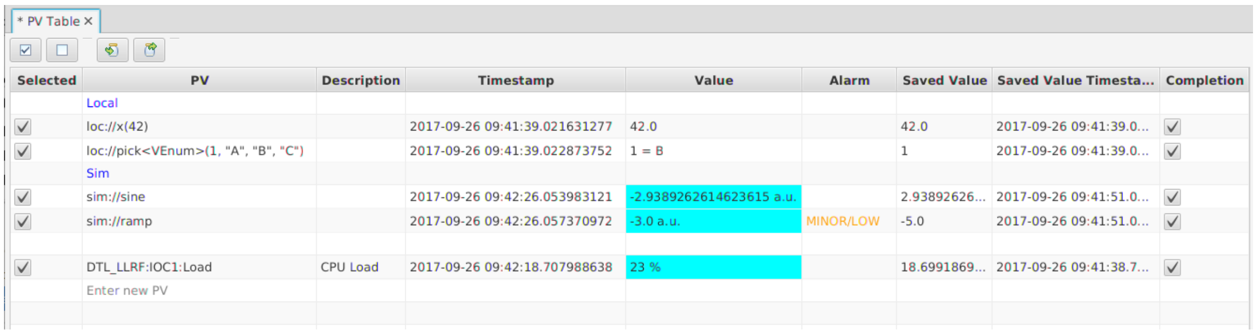
The configuration (PV names, saved value, saved date) can be saved and later re-loaded, see details on the file format described below.

### 8.25.1 Adding and Removing PVs

The simplest way is to enter new PV names in last row of table.

To insert a new PV in the middle of the table, open the context menu on the desired table row and select "Insert Row (above)" to add a row above, then change the PV name of the new row.

Finally, you use drag-and-drop to move existing rows within the table, or to 'drop' PV names into the table from tools that support dragging PV names.



| Selected | PV | Description | Timestamp | Value | Alarm | Saved Value | Saved Value Timesta... | Completion |
|---|---|---|---|---|---|---|---|---|
| | Local | | | | | | | |
| ✓ | loc://x(42) | | 2017-09-26 09:41:39.021631277 | 42.0 | | 42.0 | 2017-09-26 09:41:39.0... | ✓ |
| ✓ | loc://pick<VEnum>(1, "A", "B", "C") | | 2017-09-26 09:41:39.022873752 | 1 = B | | 1 | 2017-09-26 09:41:39.0... | ✓ |
| | Sim | | | | | | | |
| ✓ | sim://sine | | 2017-09-26 09:42:26.053983121 | -2.9389262614623615 a.u. | | 2.93892626... | 2017-09-26 09:41:51.0... | ✓ |
| ✓ | sim://ramp | | 2017-09-26 09:42:26.057370972 | -3.0 a.u. | MINOR/LOW | -5.0 | 2017-09-26 09:41:51.0... | ✓ |
| | | | | | | | | |
| ✓ | DTL_LLRF:IOC1:Load | CPU Load | 2017-09-26 09:42:18.707988638 | 23 % | | 18.6991869... | 2017-09-26 09:41:38.7... | ✓ |
| | Enter new PV | | | | | | | |

Fig. 2: PV Table

Delete PV names by changing their name to an empty name, or by selecting one or more PVs and deleting them via the context menu.

## 8.25.2 Comments

PV names that start with "# " are considered a comment. This can also be used to add empty lines into the table by entering just "#" as a PV name.

## 8.25.3 Checking PVs for Snapshot/Restore

By default, the check mark at the start of each table row is set. When taking a snapshotData of current values or restoring PVs to the snapshotData, this typically applies to rows where the check mark is set.

You can un-check table rows if they should be excluded. The context menu of the table offers shortcuts to select or de-select the whole table.

In addition, the context menu also allows taking a snapshotData or restoring the row on which the context menu was invoked, which can be useful to operate on just one PV and not the whole table.

## 8.25.4 Restoring PVs

The value of PVs can be restored, i.e. the saved value will be written to the PV. By default, this affects every row of the table, but the check-mark at the start of each table row can be use to de-select rows.

## 8.25.5 Completion

By default, saved values are restored to PVs by simply writing to them. When checkbox in the "Completion" column is selected for an PV, the saved value will be restored by using the "Put-Callback" method of writing, awaiting the completion of the write. This can be useful with PVs that support put-callback, a typical example being motors.

The PV Table has one global timeout that is used for each write operation that uses completion. It defaults to 60 seconds and can be changed via the "Completion Timeout" option in the context menu.

## 8.25.6 Tolerance

Values are highlighted when they differ from the saved snapshotData value by a certain amount. The currently used tolerance is displayed in the tool-tip of a table row. This 'tolerance' value can be configured via the context menu of selected table rows.

When configuring the 'tolerance', note that it applies to the rows which are selected in the table via the usual selection mechanism (click on one row, shift-click to select multiple rows, . . . ). If no row specific rows are selected to set their tolerance, the tolerance for every row in the table will be updated. This is independent of the check mark in the first table column which marks rows to be restored by writing their saved value back to the PVs.

### 8.25.7 File Formats

The original PVTable file format uses a ".pvs" extension for its file names. The files have an XML format which is described by the <file>pv_table.xsd</file> contained in the PV Table sources.

Since version 4.0.0, the PVTable also supports file format used by the EPICS synApps `autosave` module, http://www.aps.anl.gov/bcda/synApps/autosave/autosave.html. Whenever loading or saving a PVTable from a file with a ".sav" extension, the autosave format will be used.

Advantages of the original PVTable ".pvs" file format:

- Tracks which rows were selected.
- Saves not only the value but also the timestamp of saved values.
- Contains global as well as per-element 'tolerance'.
- Allows using 'completion'.
- Best for standalone operation of the PVTable.

Advantages of the autosave ".sav" file format:

- It can be used by the IOC to load/save settings.
- PVTable allows easy comparison of last settings written by IOC against current values.
- Best for use together with on-demand save/restore on the IOC.

## 8.26 Olog

Olog is an electronic logbook client for the logbook service maintained here: https://github.com/Olog/olog-es.

**NOTE**: this is an optional module. For information on how to build a site specific product, see https://github.com/ControlSystemStudio/phoebus/tree/master/phoebus-product.

### 8.26.1 Features

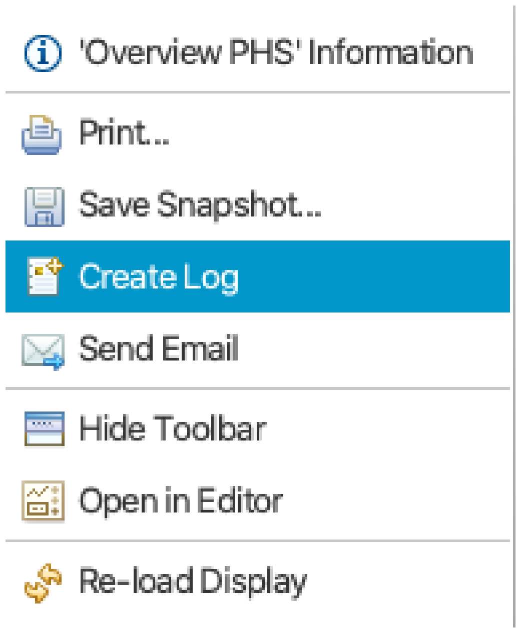
- Arbitrary number of "logbooks", configured in the service. A logbook entry is contained in one or several logbooks.
- Arbitrary number of "tags", configured in the service. A logbook entry may be associated with zero or several tags.
- Arbitrary number of "properties", configured on the service. A property is a named list of key/value pairs. The user may define values for the items in a property. A logbook entry is associated with zero or several properties.
- Arbitrary number of attachments, i.e. images or other file types.
- Markup as defined by the Commonmark specification (https://commonmark.org).
- Log entry editor invocation from context menu whereby context specific attachments or data are automatically appended to the log entry.
- Log entry viewers offer search capabilities based on meta data and content.

### 8.26.2 Launching the log entry editor

The log entry editor is launched as a non-modal window using one of the following methods:

- From the dedicated button in the application toolbar.
- From application menu Applications -> Utility -> Send to Logbook.
- Using the New Log Entry button in the log entry details view of the logbook application.
- Using the New Log Entry context menu item in the search result list view of the logbook application. This option also supports the keyboard combination CTRL+N.

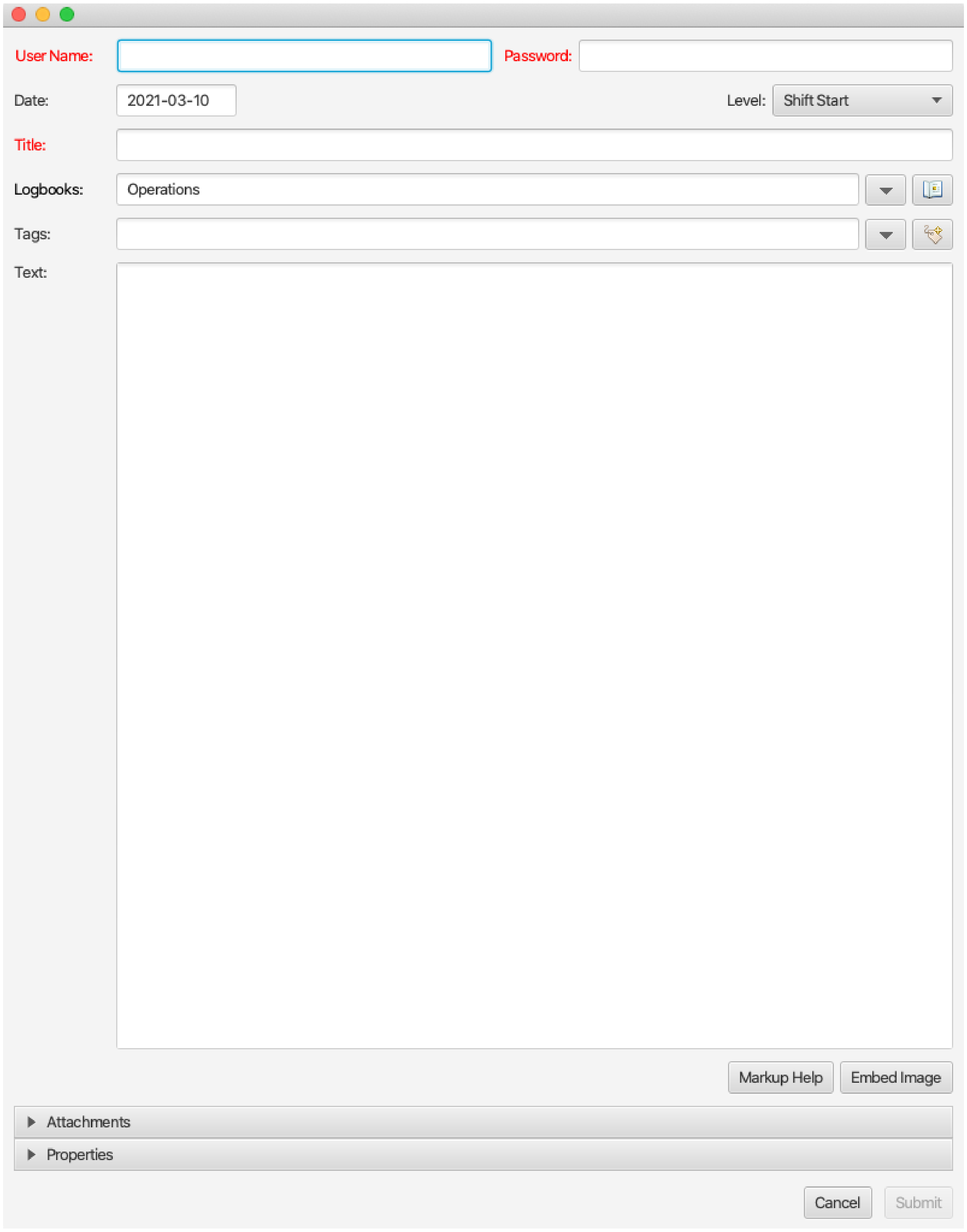
The log entry editor may also be launched from context menus, where applicable. For instance, with a right click on the background of an OPI the launched context menu will include the Create Log item:



The Create Log context menu item is available also in a Databrowser plot area.

### 8.26.3 Editing a log entry

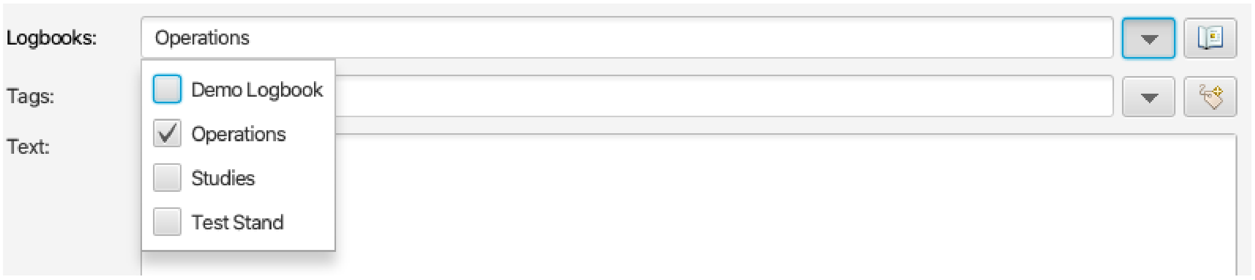
The log entry editor is a non-modal dialog:


User Name: Password:
Date: 2021-03-10 Level: Shift Start
Title:
Logbooks: Operations
Tags:
Text:

Mandatory data are:

- Username and password, see also *preferences*.

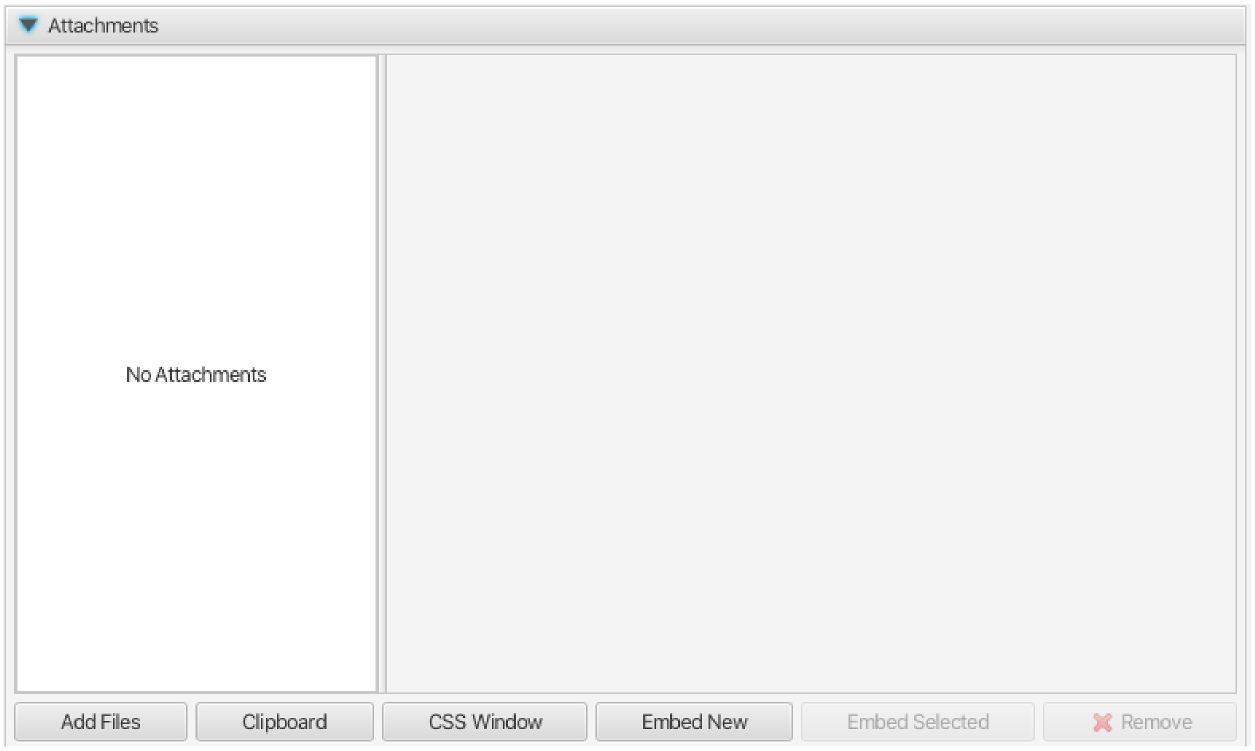
- Title
- At least one logbook, see also *preferences*. Additional logbooks - configured in the service - can be added from a list shown when pressing the down button:



The body text of the log entry can be styled using markup as defined by the Commonmark specification (https://commonmark.org). The Markup Help button will launch the system default browser to display a quick reference.

## 8.26.4 Attachments

When the log entry editor is launched from a context menu, a screen shot is automatically appended, where applicable. Additional images (or other type of attachments) may be added by expanding the Attachments editor:



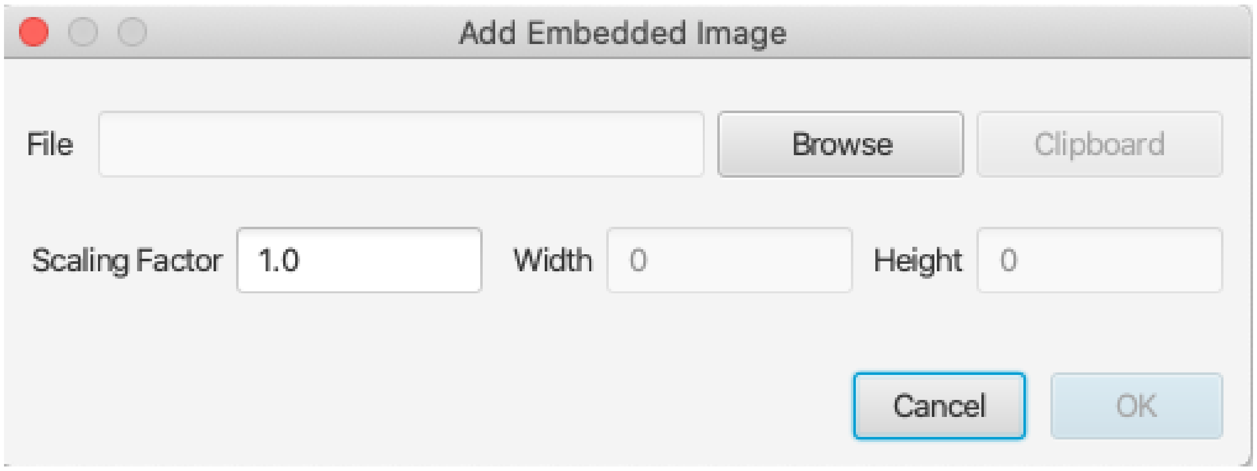
Here user may attach any number of files of arbitrary types:

- `Add Files` will launch the native file browser dialog from which user may select any number of files.
- `Clipboard` will attach the file - if any - currently copied to the host OS clipboard.
- `CSS Window` will attach an image of the current application window.
- `Embed New` will launch the dialog to embed an image to the log entry body, see below.
- `Embed Selected` will embed user selected image files previously added to the list of attachments.

**NOTE**: The Olog service will not accept upload of attachments larger than the configured limit of 50MB. The Olog service can be configured to use a different limit, but users should keep in mind that download of large attachments to the log viewer may incur delays in terms of UI updates.

### 8.26.5 Embedded images

Images may be embedded in the body text using markup. The user should consult the quick reference (Markup Help button) for details on how to do this. In general, users should use the Embed Image button to add image markup at the cursor position:



Scaling Factor 1.0 Width 0 Height 0

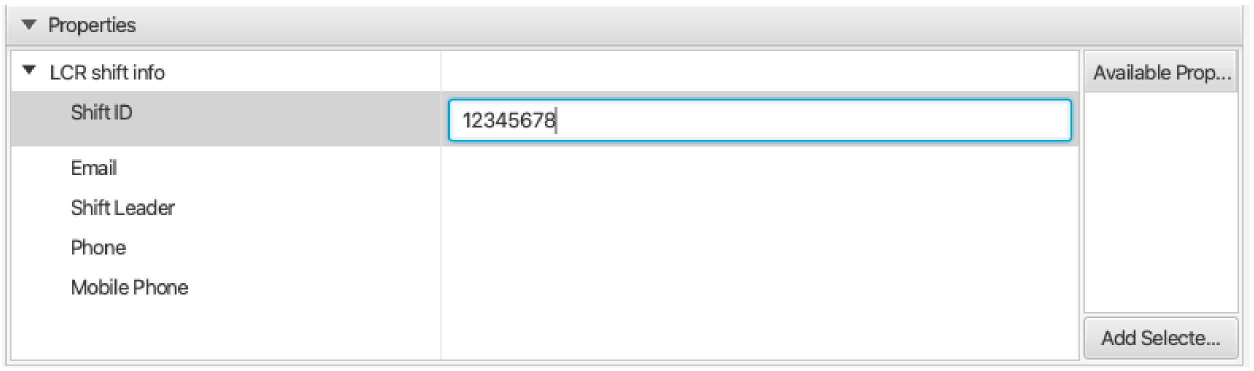
External image resources may be edited manually, e.g.: `![alt-text](https://foo.com/bar.jpg)`. File URLs are not supported.

### 8.26.6 Links

Links contained in a log entry will be opened in the default browser rather than in the view showing the log entry.

### 8.26.7 Properties

Properties are edited by expanding the Properties editor. The below screen shot shows that one single property (LCR shift info) holding five keys has been configured in the service:

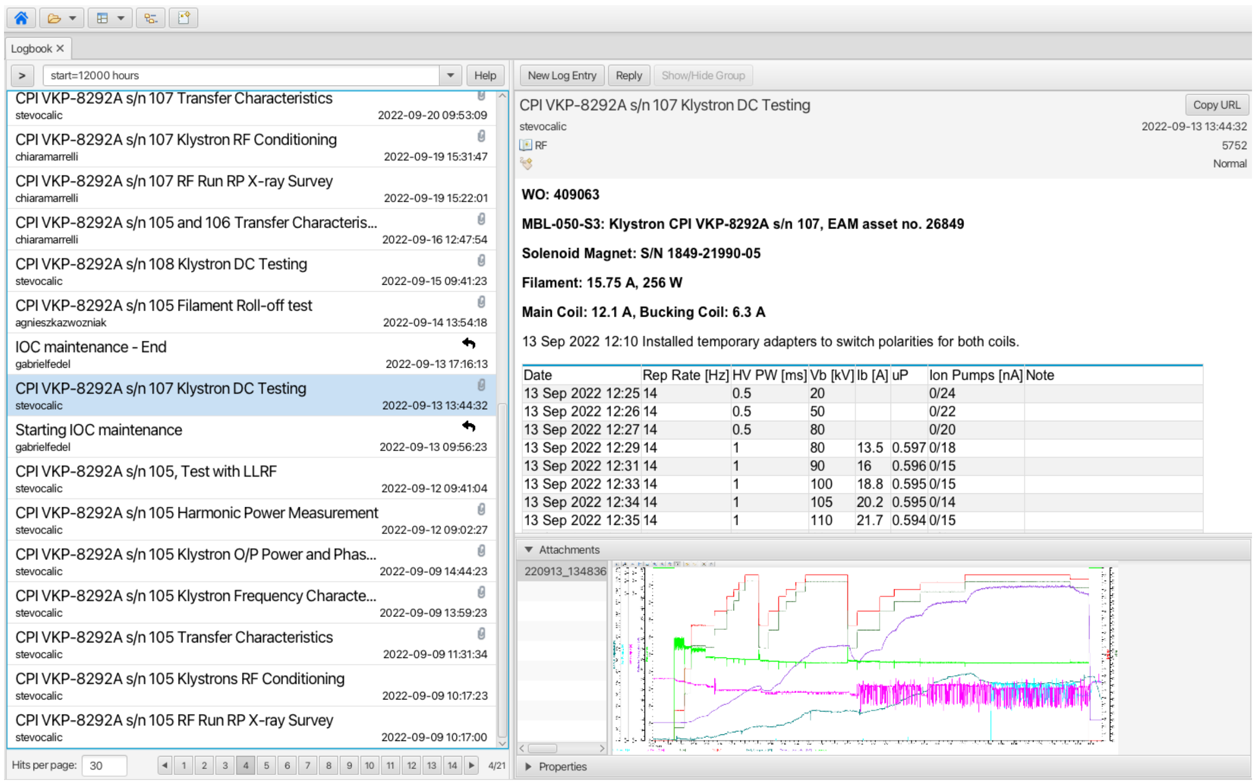


| LCR shift info | |
|---|---|
| Shift ID | 12345678 |
| Email | |
| Shift Leader | |
| Phone | |
| Mobile Phone | |

User may select what properties to include in the log entry, and edit the values for the items in the property.

### 8.26.8 Log entry viewer

The menu item Applications -> Utility -> Log Entry Table will launch an application (in a new tab) in which the user may search and view log entries:



start=12000 hours

CPI VKP-8292A s/n 107 Transfer Characteristics
stevocalic 2022-09-20 09:53:09

CPI VKP-8292A s/n 107 Klystron RF Conditioning
chiaramarrelli 2022-09-19 15:31:47

CPI VKP-8292A s/n 107 RF Run RP X-ray Survey
chiaramarrelli 2022-09-19 15:22:01

CPI VKP-8292A s/n 105 and 106 Transfer Characteris...
chiaramarrelli 2022-09-16 12:47:54

CPI VKP-8292A s/n 108 Klystron DC Testing
stevocalic 2022-09-15 09:41:23

CPI VKP-8292A s/n 105 Filament Roll-off test
agnieszkazwozniak 2022-09-14 13:54:18

IOC maintenance - End
gabrielfedel 2022-09-13 17:16:13

CPI VKP-8292A s/n 107 Klystron DC Testing
stevocalic 2022-09-13 13:44:32

Starting IOC maintenance
gabrielfedel 2022-09-13 09:56:23

CPI VKP-8292A s/n 105, Test with LLRF
stevocalic 2022-09-12 09:41:04

CPI VKP-8292A s/n 105 Harmonic Power Measurement
stevocalic 2022-09-12 09:02:27

CPI VKP-8292A s/n 105 Klystron O/P Power and Phas...
stevocalic 2022-09-09 14:44:23

CPI VKP-8292A s/n 105 Klystron Frequency Characte...
stevocalic 2022-09-09 13:59:23

CPI VKP-8292A s/n 105 Transfer Characteristics
stevocalic 2022-09-09 11:31:34

CPI VKP-8292A s/n 105 Klystrons RF Conditioning
stevocalic 2022-09-09 10:17:23

CPI VKP-8292A s/n 105 RF Run RP X-ray Survey
stevocalic 2022-09-09 10:17:00

Hits per page: 30

CPI VKP-8292A s/n 107 Klystron DC Testing
stevocalic 2022-09-13 13:44:32
RF 5752
Normal

**WO: 409063**

**MBL-050-S3: Klystron CPI VKP-8292A s/n 107, EAM asset no. 26849**

**Solenoid Magnet: S/N 1849-21990-05**

**Filament: 15.75 A, 256 W**

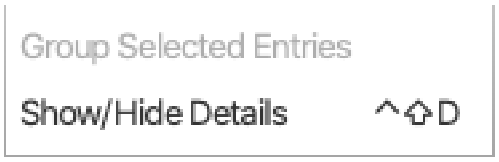
**Main Coil: 12.1 A, Bucking Coil: 6.3 A**

13 Sep 2022 12:10 Installed temporary adapters to switch polarities for both coils.

| Date | Rep Rate [Hz] | HV PW [ms] | Vb [kV] | Ib [A] | uP | Ion Pumps [nA] | Note |
|---|---|---|---|---|---|---|---|
| 13 Sep 2022 12:25 | 14 | 0.5 | 20 | | | 0/24 | |
| 13 Sep 2022 12:26 | 14 | 0.5 | 50 | | | 0/22 | |
| 13 Sep 2022 12:27 | 14 | 0.5 | 80 | | | 0/20 | |
| 13 Sep 2022 12:29 | 14 | 1 | 80 | 13.5 | 0.597 | 0/18 | |
| 13 Sep 2022 12:31 | 14 | 1 | 90 | 16 | 0.596 | 0/15 | |
| 13 Sep 2022 12:33 | 14 | 1 | 100 | 18.8 | 0.595 | 0/15 | |
| 13 Sep 2022 12:34 | 14 | 1 | 105 | 20.2 | 0.595 | 0/14 | |
| 13 Sep 2022 12:35 | 14 | 1 | 110 | 21.7 | 0.594 | 0/15 | |

220913_134836

User may choose to hide some details of each log entry in the list in order to fit more items in the view and to reduce the need for scrolling. This can be done using the keyboard shortcut `CTRL+SHIFT+D`, or by selecting the `Show/Hide Details` item from the context menu invoked through a right click in the table view. The choice to show or hide details is persisted between restarts of the application.


Group Selected Entries

Show/Hide Details ^⇧D

In the search field the user may specify criteria when searching for log entries. These criteria are based on the elements of a log entry as follows:

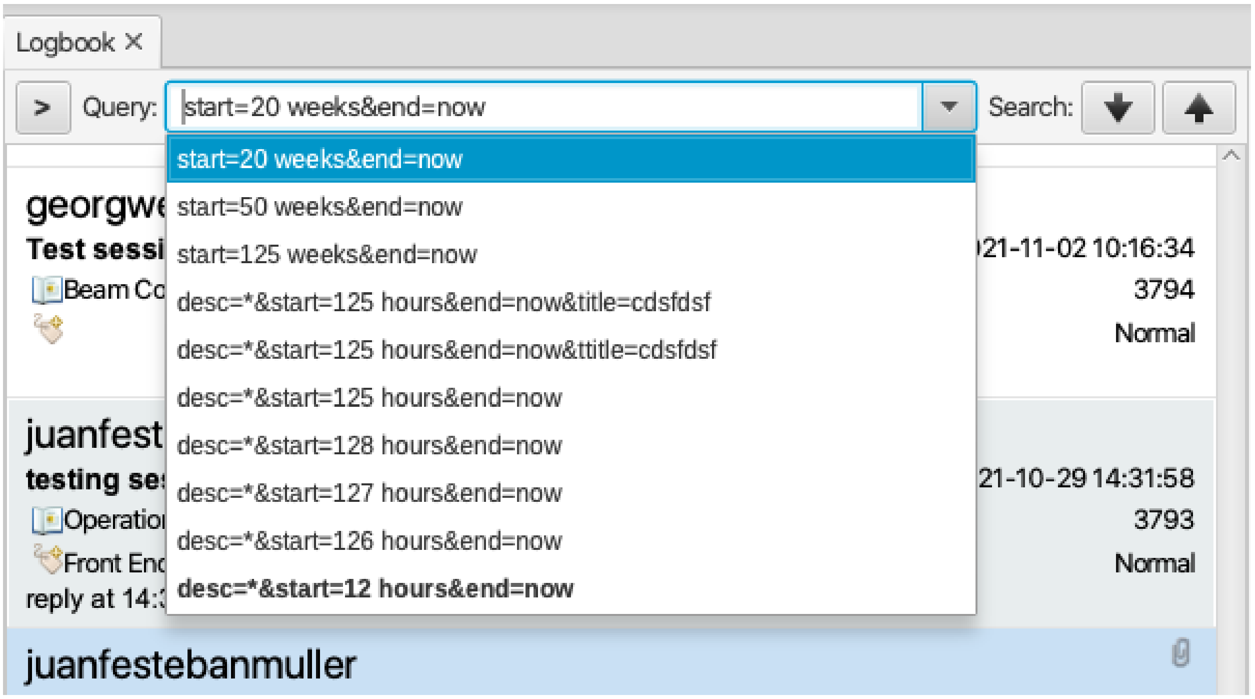
- `desc` or `description`: The body text, whereby any markup characters are ignored. The search is case insensitive.
- `title`: The title of the log entry. The search is case insensitive.
- `level`: The value of the Level field.
- `logbooks`: A comma separated list of logbook names. Log entries contained in either of the listed logbooks will match.
- `tag`: A comma separated list of tag names. Log entries tagged with either of the listed tags will match.
- `owner`: The author of a log entry as specified in the Username field when the entry was created.

- `start`: Defines the start date limit in a search. Time may be specified using the format `yyyy-MM-dd HH:mm:ss.SSS` or a relative time like “8 hours” or “2 days”.
- `end`: Defines the end date limit in a search. Time may be specified using the format `yyyy-MM-dd HH:mm:ss.SSS` or a relative time like “8 hours” or “2 days”. The value “now” is supported.
- `properties`. Both property names as well as key name and value of the items in a property can be searched like so:
  - `properties=property name` find log entries containing a property named “property name”
  - `properties=property name.key name` find log entries containing a property names “property name” and that contains a key named “key name”.
  - `properties=property name.key name.value` find log entries containing a property named “property name” and that contains a key named “key named” with a value of “value”.
  - `properties=property name 1|property name 2` find log entries containing a property named “property name 1” **or** a property named “property name 2”. The pipe character is used to separate search expressions.

### Query history

Search queries entered by the user are put onto a first-in-first-out query history list. A button next to the search field will expand a drop-down box to show previously used queries, see screen shot below. Queries are ordered by last-used-time where the most recent query is on top. When new queries are entered by user, older queries may be flushed out as the maximum size of the list is limited (15 by default, configurable between 5 and 30). The “default” search query - rendered in bold font in the list - as defined in the preferences is however never flushed.

When user has selected a query from the list, a search button (up or down arrow) must be clicked in order to dispatch the search request. Pressing ENTER when editing a query in the search field will also trigger a search, and the query is put in the history list.

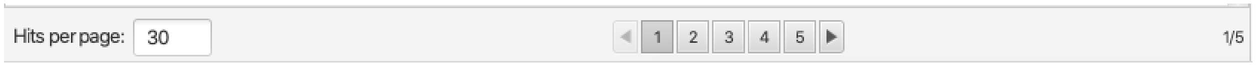


start=20 weeks&end=now

start=20 weeks&end=now
start=50 weeks&end=now
start=125 weeks&end=now
desc=*&start=125 hours&end=now&title=cdsfdsf
desc=*&start=125 hours&end=now&ttitle=cdsfdsf
desc=*&start=125 hours&end=now
desc=*&start=128 hours&end=now
desc=*&start=127 hours&end=now
desc=*&start=126 hours&end=now
**desc=*&start=12 hours&end=now**

### Pagination

Each search request will retrieve a limited number of matching log entries to render in the list view. This limit - aka "page size" - defaults to 30, but may be changed by a property value override. In addition, user may override the default page size in the UI. Page size must be between 1 and 999. If the search results in a hit count larger than the page size, the UI will render page navigation buttons below the list of log entries. The current page and total number of pages is also shown, see screen shot. The navigation buttons are not rendered if hit count less or equal to the page size.



### Periodic Search

When a user-initiated search request has completed, a background task is launched to repeatedly (once every 30 seconds) perform a new search using the same search query. If the user edits the query to launch a new search request, the current periodic search is aborted and re-launched when the search request completes.

The periodic search feature will consequently keep the list of matching queries updated when new log entries matching the current query are added.

Any failure in a search request - whether manually triggered by the user or by the background task - will abort the periodic search. User will need to trigger another search request to restart the process.

## 8.26.9 Attachment Preview

When viewing a log entry, attachments are listed in the attachments view. A preview of an image attachment is shown when user selects it. To see the attachment in full resolution, user may click on the preview image, or double-click in the attachment list.

If user double-clicks on a OPI file attachment (.bob file), the application will launch that OPI in run mode.

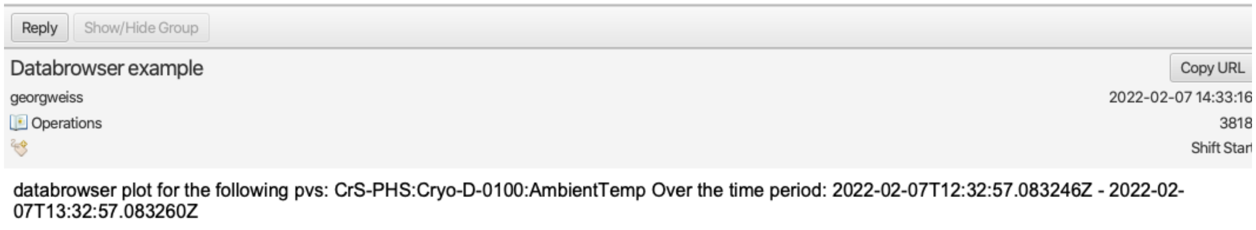
If user double-clicks on a Data Browser attachment (.plt file), the application will launch the Data Browser.

Preview of non-image files is not offered in the application. However, external viewers may be configured for arbitrary file extensions, see **preference_settings_** (framework.workbench) for more information.

## 8.26.10 Log Entry Grouping

The preference setting `log_entry_groups_support` - if set to `true` - will enable the "log entry grouping" feature. With this users will be able to reply to individual log entries implicitly creating a group of log entries. To use this feature user can choose to:

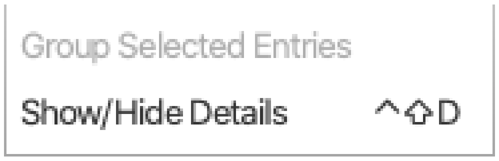
- Press the Reply button shown in the log entry view:



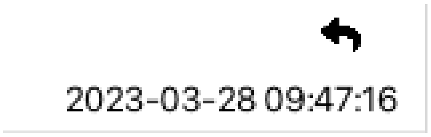
Databrowser example
georgweiss
Operations
2022-02-07 14:33:16
3818
Shift Start

databrowser plot for the following pvs: CrS-PHS:Cryo-D-0100:AmbientTemp Over the time period: 2022-02-07T12:32:57.083246Z - 2022-02-07T13:32:57.083260Z

- Select "Group Selected Entries" from the context menu shown on right click in the search result table view. This menu item is enabled when at least two items are selected:



Log entries that are contained in a log entry group are rendered with a “reply” icon in the search result table view:



In the log entry view, the “Show/Hide Group” button (see screen shot above) can be used to show all log entries of a group sequentially, ordered on created date with oldest log entry on top. In this merged view attachments and properties are not shown. Clicking on a header in the merged view will show that log entry in full.

**NOTE**: To be able to group log entries user must be authenticated in one of the following manners:

- Use “credentials caching” through preference setting `org.phoebus.logbook.olog.ui/save_credentials`. Once a log entry has been created, credentials will be reused when creating a group.
- Use the Credentials Management app to sign in to the logbook context.

#### Limitations

Please consider the following limitations of the log entry group feature:

- A log entry group should not be regarded as a discussion thread.
- There is no support for “groups of groups”, or “sub-groups”.
- There is no parent-child relation between log entries in a group, i.e. there is no internal structure of the log entries in a group.
- A log entry may be included in only one log entry group. It is hence not possible to create a new group of log entries if these are already contained in different groups.

### 8.26.11 Preferences

Preferences related to the electronic logbook are the following:

- `org.phoebus.olog.es.api/olog_url`. This should be on the format `http(s)://foo.com/Olog`, where the path element `Olog` may not be omitted.
- `org.phoebus.logbook.olog.ui/default_logbooks`. This is a comma separated list of logbooks automatically associated with a new log entry.
- `org.phoebus.logbook.olog.ui/level_field_name`. The text shown next to the drop-down below the password field. Sites may wish to customize this to override the default value “Level”.
- `org.phoebus.olog.es.api/levels`. List of items shown in the “Level” drop-down.
- `org.phoebus.logbook.ui/save_credentials`. Indicates if user credentials should be cached. If `true`, the user will have to specify credentials only for the first new log entry after launch of CS Studio. The side effect of credentials caching is that all entries will be created with the same user (owner) identity.
- `search_result_page_size`. The maximum number of hits per page to fetch and render in a search. User may override in the UI. Value must be 1 - 999, default is 30.

- `log_entry_groups_support`. If true, user may reply to log entries and create a log entry group from a selection of existing log entries.

# CHAPTER 9

# Services

The following sections describe available services.

## 9.1 Save-and-restore service

The save-and-restore service implements service as a collection of REST endpoints. These can be used by clients to manage configurations (aka save sets) and snapshots, to compare snapshots and to restore PV values from snapshots.

The service is packaged as a self-contained Spring Boot jar file. External dependencies are limited to a JVM (Java 11+) and a running instance of Elasticsearch (8.x).

### 9.1.1 Running the service

The file `application.properties` lists a few settings that can be customized to each site's need, e.g. connection parameters for Elasticsearch.

### 9.1.2 Elasticsearch setup

There is no need to manually created the Elasticsearch indices as these are created by the application if they do not yet exist.

## 9.2 REST API for Save Restore Service

### 9.2.1 Node

Data is arranged such that is can be rendered in a tree structure, where each node is of a specific type. See below for details. A root node is always available and cannot be deleted.

Each node is uniquely identified through an UUID id. The root node's unique id is always `44bef5de-e8e6-4014-af37-b8f6c8a939a2`.

REST end-points documented below can be used to locate particular nodes, or traverse the tree by listing child nodes.

### 9.2.2 Node types

**Folder:**

A folder node is a container for folder and configuration nodes. The root node is a folder node.

**Configuration:**

A configuration node is essentially a set of PVs defining what data to put in a snapshot. Configuration nodes must be created in folder nodes, though not in the root node.

For each such PV "item" one may also specify:

- a read-back PV
- flag to indicate if the PV should restored in a restore operation

**Snapshot:**

A snapshot node consists of a list of PV values at a particular instant in time. To take a snapshot the client must point to a configuration defining this list of PVs (and optionally read-back PVs). In other words, when saving a snapshot the client must specify the unique id of the associated configuration node.

**Composite Snapshot:**

An aggregation of snapshot nodes and/or other composite snapshot nodes. The referenced nodes must exist in order to be able to create a composite snapshot. Moreover, a snapshot node cannot be deleted if it is referenced in a composite snapshot.

### 9.2.3 REST Services

The service is implemented as a REST style web service, which – in this context – means:

- The URL specifies the data element that the operation works upon.
- The HTTP method specifies the type of operation.

GET: retrieve an element, does not modify data
PUT: create an element
POST: update the addressed element
DELETE: delete the addressed element

### 9.2.4 Node Management

#### Get a node

**.../node/{uniqueNodeId}**

Method: GET

Return: The details of the node with id *{uniqueNodeId}*

```
{
    "uniqueId": "ae9c3d41-5aa0-423d-a24e-fc68712b0894",
    "name": "CSX",
    "created": 1623701056000,
    "lastModified": 1623780701000,
    "nodeType": "FOLDER",
    "userName": "kunal",
    "tags": []
}
```

Nodes of type CONFIGURATION and SNAPSHOT will also have a `description` field.

A special case is the root node as it has a fixed unique id:

**…/node/44bef5de-e8e6-4014-af37-b8f6c8a939a2**

### Create a new node

**…/node?parentNodeId=<parent's node id>**

Method: PUT

Body:

```
{
    "name": "New_Node_Camera",
    "nodeType": "CONFIGURATION",
    "userName": "kunal"
}
```

nodeType: "CONFIGURATION" or "FOLDER". The request parameter `parentNodeId` is mandatory and must identify an existing folder node.

The nodeType can be used to specify if we want to create a new folder or a new configuration.

Return: If the node was successfully created you will a 200 response with the details of the newly created node

```
{
    "uniqueId": "c4302cfe-60e2-46ec-bf2b-dcd13c0ef4c0",
    "name": "New_Node_Camera",
    "created": 1625837873000,
    "lastModified": 1625837873000,
    "nodeType": "CONFIGURATION",
    "userName": "kunal",
    "tags": []
}
```

### Update a node

**…/node**

Method: POST

Return: The updated node.

```
{
    "uniqueId": "ae9c3d41-5aa0-423d-a24e-fc68712b0894",
    "name": "new name",
    "description": "new description",
    "created": 1623701056000,
    "lastModified": 1623780701000,
    "nodeType": "CONFIGURATION",
    "userName": "kunal",
    "tags": []
}
```

Updates an existing node with respect to its name or description, or both. The `nodeType` cannot be updated.

### Delete a node

**. . . /node/{uniqueNodeId}**

Method: DELETE

Deletes the node identified by `uniqueNodeId`. Deletion is agnostic to the node type.

Note that deletion is recursive:

- Deleting a configuration node will also delete all associated snapshot nodes.
- Deleting a folder node will delete also delete all nodes in its sub-tree.

### Get a node parent

**. . . /node/{uniqueNodeId}/parent**

Method: GET

Return: The details of the *parent* node of the node with id *{uniqueNodeId}*

### Get children

**. . . /node/{uniqueNodeId}/children**

Method: GET

Return: The a list of all the children nodes of the node with id *{uniqueNodeId}*

```
[
    {
        "uniqueId": "8cab9311-0c77-4307-a508-a33677ecc631",
        "name": "Camera",
        "created": 1623701073000,
        "lastModified": 1625836981000,
        "nodeType": "CONFIGURATION",
        "userName": "kunal",
        "tags": []
    },
    {
        "uniqueId": "3aa5baa3-8386-4a74-84bb-5fdd9afccc7f",
        "name": "ROI",
        "created": 1623780701000,
```

(continues on next page)

(continued from previous page)

```
        "lastModified": 1623780701000,
        "nodeType": "CONFIGURATION",
        "userName": "kunal",
        "tags": []
    }
]
```

### Get a configuration

To get a configuration node the client should call the end-point associated with getting nodes of any type:

**.../node/{uniqueNodeId}**

where `uniqueNodeId` identifies the configuration node.

The actual configuration data associated with a configuration node is maintained in a separate Elasticsearch index and is accessible through:

**.../config/{uniqueNodeId}**

where `uniqueNodeId` identifies the configuration node.

Method: GET

Return: object describing the configuration data, essentially a list of PVs.

```
{
    "uniqueId": "89886b32-bb2e-4336-8eea-375c0a955cad",
    "pvList": {
        [
            {
                "pvName": "13SIM1:{SimDetector-Cam:1}cam1:BinX"
            },
            {
                "pvName": "13SIM1:{SimDetector-Cam:1}cam1:BinY"
            },
            {
                "pvName": "13SIM1:{SimDetector-Cam:2}cam2:BinX",
                "readbackPvName": null,
                "readOnly": false
            },
            {
                "pvName": "13SIM1:{SimDetector-Cam:2}cam2:BinY",
                "readbackPvName": null,
                "readOnly": false
            }
        ]
    }
}
```

Here the `uniqueId` field matches the `unqiueId` field of the configuration node.

### Create a configuration

**.../config?parentNodeId=<parent's node id>**

Method: PUT

Return: an object representing the saved configuration. This object is of the same type as the body sent in the request, with additional data set by the service, e.g. the unique id of the created configuration node.

Body:

```
{
    "configurationNode": {
         "name": "New_Configuration",
         "nodeType": "CONFIGURATION",
         "userName": "kunal"
    },
    "configurationData": {
        "pvList": {
            [
                {
                    "pvName": "13SIM1:{SimDetector-Cam:1}cam1:BinX"
                },
                {
                    "pvName": "13SIM1:{SimDetector-Cam:1}cam1:BinY"
                },
                {
                    "pvName": "13SIM1:{SimDetector-Cam:2}cam2:BinX",
                    "readbackPvName": null,
                    "readOnly": false
                },
                {
                    "pvName": "13SIM1:{SimDetector-Cam:2}cam2:BinY",
                    "readbackPvName": null,
                    "readOnly": false
                }
            ]
        }
    }
}
```

The request parameter `parentNodeId` is mandatory and must identify an existing folder node. The client needs to specify a name for the new configuration node, as well as a user identity.

#### Update a configuration

**.../config/{uniqueNodeId}**

Method: POST

This endpoint works in the same manner as the for the PUT method, i.e. the body and return value are the same. However, in this case the `uniqueNodeId` must identify an existing configuration node.

The body can specify a new name or description, or both. On top of that the list of PVs can be updated. It should be noted though that the specified list will replace the existing one, i.e. all PVs that must remain in the updated configuration data must be listed in the body. Any PVs in the existing configuration data missing from the body will be removed.

### 9.2.5 Snapshot Endpoints

#### Get a snapshot

To get a snapshot node the client should call the end-point associated with getting nodes of any type:

**.../node/{uniqueNodeId}**

where `uniqueNodeId` identifies the snapshot node.

The actual snapshot data associated with a snapshot node is maintained in a separate Elasticsearch index and is accessible through:

**.../snapshot/{uniqueNodeId}**

where `uniqueNodeId` identifies the snapshot node.

Method: GET

Return: object describing the snapshot data, essentially a list of PVs and the persisted values.

```
{
    "uniqueId":"54920ffe-8932-46e6-b420-5b7b20d2cea1",
    "snapshotItems":[
        {
            "configPv": {
                "pvName":"COUNTER10",
                "readOnly":false
            },
            "value":{
                "type":{
                    "name":"VDouble",
                    "version":1
                },
                "value":11941.0,
                "alarm":{
                    "severity":"NONE",
                    "status":"NONE",
                    "name":"NO_ALARM"
                },
                "time":{
                    "unixSec":1664550284,
                    "nanoSec":870687555
                },
                "display":{
                    "lowDisplay":0.0,
                    "highDisplay":0.0,
                    "units":""
                }
            }
        },
        {
            "configPv":{
                "pvName":"TEMP10",
                "readOnly":false
            },
            "value":{
                "type":{
                    "name":"VDouble",
                    "version":1
                },
                "value":-4.205873713538651,
                "alarm":{
                    "severity":"MINOR",
                    "status":"NONE",
                    "name":"LOW_ALARM"
```

(continues on next page)

(continued from previous page)

```
                },
                "time":{
                    "unixSec":1664550284,
                    "nanoSec":870768480
                },
                "display":{
                    "lowAlarm":-5.0,
                    "highAlarm":30.0,
                    "lowDisplay":-60.0,
                    "highDisplay":60.0,
                    "lowWarning":0.0,
                    "highWarning":10.0,
                    "units":"°"
                }
            }
        }
    ]
}
```

To be noted: the `value` field is a serialized version of the underlying EPICS PV objects. The contents of this field will hence depend on the EPICS record type and its properties.

## Save a snapshot

**.../snapshot?parentNodeId=<parent's node id>**

Method: PUT

Return: an object representing the saved snapshot. This object is of the same type as the body sent in the request, with additional data set by the service, e.g. the unique id of the created snapshot node.

Body:

```
{
    "snapshotNode": {
         "name": "New_Snapshot",
         "nodeType": "SNAPSHOT",
         "userName": "kunal"
    },
    "snapshotData": {
        "snapshotItems":[
            {
                "configPv": {
                    "pvName":"COUNTER10",
                    "readOnly":false
                },
                "value":{
                    "type":{
                        "name":"VDouble",
                        "version":1
                    },
                    "value":11941.0,
                    "alarm":{
                        "severity":"NONE",
                        "status":"NONE",
                        "name":"NO_ALARM"
```

(continues on next page)

(continued from previous page)

```
                    },
                    "time":{
                        "unixSec":1664550284,
                        "nanoSec":870687555
                    },
                    "display":{
                        "lowDisplay":0.0,
                        "highDisplay":0.0,
                        "units":""
                    }
                }
            },
            {
                "configPv":{
                    "pvName":"TEMP10",
                    "readOnly":false
                },
                "value":{
                    "type":{
                        "name":"VDouble",
                        "version":1
                    },
                    "value":-4.205873713538651,
                    "alarm":{
                        "severity":"MINOR",
                        "status":"NONE",
                        "name":"LOW_ALARM"
                    },
                    "time":{
                        "unixSec":1664550284,
                        "nanoSec":870768480
                    },
                    "display":{
                        "lowAlarm":-5.0,
                        "highAlarm":30.0,
                        "lowDisplay":-60.0,
                        "highDisplay":60.0,
                        "lowWarning":0.0,
                        "highWarning":10.0,
                        "units":"°"
                    }
                }
            }
        ]
    }
}
```

The request parameter `parentNodeId` is mandatory and must identify an existing configuration node. This configuration node must be the configuration node associated with the snapshot, i.e. must specify the list of PVs contained in the snapshot. The client needs to specify a name for the new snapshot node, as well as a user identity.

### 9.2.6 Composite Snapshot Endpoints

### Get a composite snapshot

To get a composite snapshot node the client should call the end-point associated with getting nodes of any type:

**.../node/{uniqueNodeId}**

where `uniqueNodeId` identifies the composite snapshot node.

The actual composite snapshot data associated with a composite snapshot node is maintained in a separate Elastic-search index and is accessible through:

**.../composite-snapshot/{uniqueNodeId}**

where `uniqueNodeId` identifies the composite snapshot node.

Method: GET

Return: object describing the composite snapshot data, essentially a list of referenced snapshot and composite snapshot nodes.

```
{
  "uniqueId": "e80fba66-c7f0-453e-8cb6-12b22fa8c957",
  "referencedSnapshotNodes": [
    "b0cee6ff-76a2-46e6-b0ef-d8b78bff26f6",
    "b6b5a03e-252e-4e6b-a9ac-9d50c23f3f0b"
  ]
}
```

### Create a composite snapshot

**.../composite-snapshot?parentNodeId=<parent's node id>**

Method: PUT

Return: an object representing the composite snapshot. This object is of the same type as the body sent in the request, with additional data set by the service, e.g. the unique id of the created composite snapshot node.

Body:

```
{
    "compositeSnapshotNode": {
         "name": "New_Composite_Snapshot",
         "nodeType": "COMPOSITE_SNAPSHOT",
         "userName": "johndoe"
    },
    "referencedSnapshotNodes": {
        [
            "b0cee6ff-76a2-46e6-b0ef-d8b78bff26f6",
            "b6b5a03e-252e-4e6b-a9ac-9d50c23f3f0b"
        ]
    }
}
```

### Update a composite snapshot

**.../composite-snapshot/{uniqueNodeId}**

Method: POST

This endpoint works in the same manner as the for the PUT method, i.e. the body and return value are the same. However, in this case the `uniqueNodeId` must identify an existing composite snapshot node.

The body can specify a new name or description, or both. On top of that the list of referenced snapshots can be updated. It should be noted though that the specified list will replace the existing one, i.e. all referenced snapshots that must remain in the updated composite snapshot data must be listed in the body. Any snapshots in the existing configuration data missing from the body will be removed.

### Get restorable items of a composite snapshot

***. . . /composite-snapshot/{uniqueId}/items**

Method: GET

Return: a list of all snapshot items as persisted in the snapshots referenced by a composite snapshot.

Body:

```
[
  {
    "configPv": {
      "pvName": "RFQ-010:RFS-EVR-101:OpMode",
      "readbackPvName": null,
      "readOnly": false
    },
    "value": {
      "type": {
        "name": "VEnum",
        "version": 1
      },
      "value": 0,
      "alarm": {
        "severity": "NONE",
        "status": "NONE",
        "name": "NONE"
      },
      "time": {
        "unixSec": 1638905851,
        "nanoSec": 445854166
      },
      "enum": {
        "labels": [
          "Global"
        ]
      }
    }
  },
  {
    "configPv": {
      "pvName": "RFQ-010:RFS-EVR-101:RFSyncDly-SP",
      "readbackPvName": null,
      "readOnly": false
    },
    "value": {
      "type": {
        "name": "VDouble",
        "version": 1
      },
```

(continues on next page)

(continued from previous page)

```
      "value": 200.0,
      "alarm": {
        "severity": "NONE",
        "status": "NONE",
        "name": "NONE"
      },
      "time": {
        "unixSec": 1638475923,
        "nanoSec": 703595298
      },
      "display": {
        "units": ""
      }
    }
  },
  {
    "configPv": {
      "pvName": "RFQ-010:RFS-EVR-101:RFSyncWdt-SP",
      "readbackPvName": null,
      "readOnly": false
    },
    "value": {
      "type": {
        "name": "VDouble",
        "version": 1
      },
      "value": 100.0,
      "alarm": {
        "severity": "NONE",
        "status": "NONE",
        "name": "NONE"
      },
      "time": {
        "unixSec": 1639063122,
        "nanoSec": 320431469
      },
      "display": {
        "units": ""
      }
    }
  },
  {
    "configPv": {
      "pvName": "RFQ-010:RFS-EVR-101:SCDly",
      "readbackPvName": null,
      "readOnly": false
    },
    "value": {
      "type": {
        "name": "VDouble",
        "version": 1
      },
      "value": 493.2,
      "alarm": {
        "severity": "NONE",
        "status": "NONE",
        "name": "NONE"
```

(continues on next page)

(continued from previous page)

```
      },
      "time": {
        "unixSec": 1639209326,
        "nanoSec": 372407313
      },
      "display": {
        "units": ""
      }
    }
  }
]
```

### 9.2.7 Migration

Commit `48e17a380b660d59b79cec4d2bd908c0d78eeeae` of the service code base is about changing the persistence component from a RDB engine to Elasticsearch. Sites using save-and-restore with an RDB engine may migrate data using the below procedure.

Terminology: "source host" is the host running the legacy service instance using a RDB engine, while "target host" is the host that will be running the updated service.

Make sure the source host is running the legacy save-and-restore service.

Make sure the target host is running the Elasticsearch service, but **not** the save-and-restore service.

On the target host, launch the save-and-restore service using the `-migrate` program argument: `java -jar /path/to/service-save-and-restore-<version>.jar -migrate http://<source host>:8080`

Here it is assumed that the legacy save-and-restore service has been published on the (default) port 8080.

If Elasticsearch is not running on localhost:9200, then add Java VM arguments like so:

`-Delasticsearch.network.host=<hostname>`

`-Delasticsearch.http.port=<port>`

## 9.3 Alarm Server

The alarm server monitors a configurable set of PVs and tracks their alarm state. When a PV goes into alarm, this is indicated in the alarm system UI. Operators will usually acknowledge the alarm to indicate that they started to investigate. They can open guidance information or related displays, and once the alarm PV recovers, the alarm clears.

For details on the original design based on JMS and an RDB, see http://accelconf.web.cern.ch/AccelConf/icalepcs2009/papers/tua001.pdf

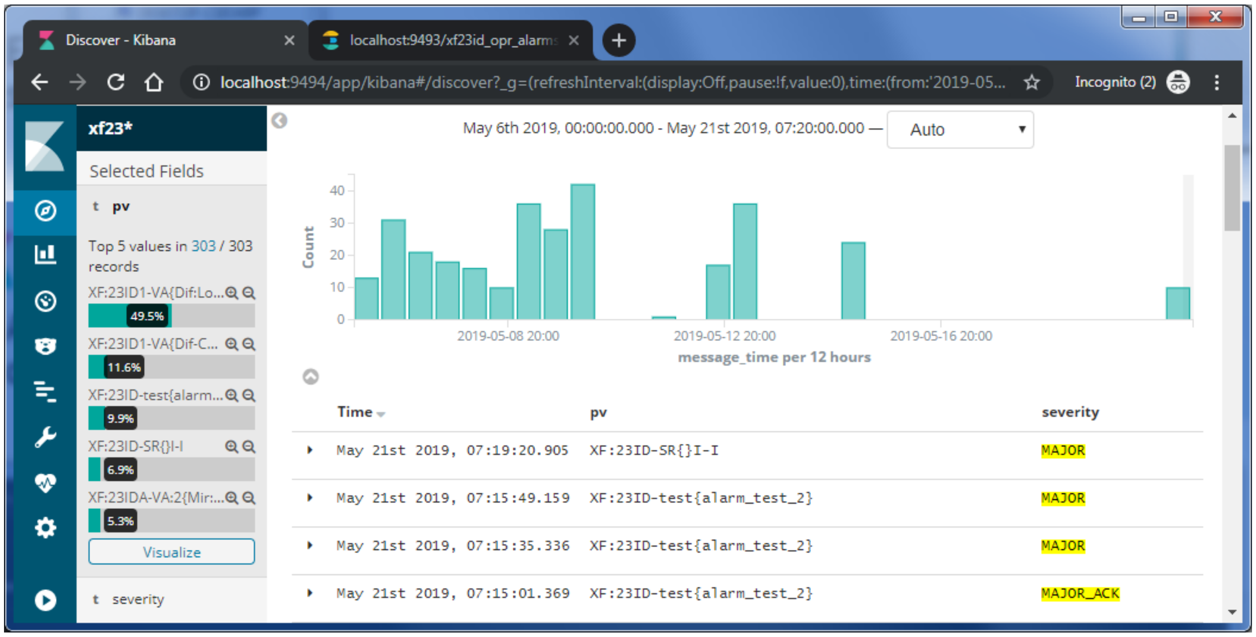
For details on setting up Kafka, see app/alarm/Readme.md

## 9.4 Alarm Logging Service

The alarm logging service records all alarm messages to create an archive of all alarm state changes and the associated actions.

This historical data can be used to:

1. Discover alarm patterns and trends
2. Generate Statistical reports on alarms
3. Debug the alarm system



| Time | pv | severity |
|---|---|---|
| May 21st 2019, 07:19:20.905 | XF:23ID-SR{}I-I | MAJOR |
| May 21st 2019, 07:15:49.159 | XF:23ID-test{alarm_test_2} | MAJOR |
| May 21st 2019, 07:15:35.336 | XF:23ID-test{alarm_test_2} | MAJOR |
| May 21st 2019, 07:15:01.369 | XF:23ID-test{alarm_test_2} | MAJOR_ACK |

### 9.4.1 Logged Alarm Messages

The alarm logging service creates kafka streams which can be configured to monitor one or more alarm topics. All associated with state change or configuration change are filtered, time stamped and added to an elastic index.

Examples:

- **Configuration changes**

  e.g. when new alarm nodes or pvs are added or removed or existing ones are enabled/disabled

- **State changes**

  e.g. alarm state changes from OK to MAJOR

- **Commands**

  e.g. a user actions to *Acknowledge* an alarm

## 9.5 RDB Archive Engine Service

The RDB archive engine reads samples from PVs and writes them to an RDB. The RDB may be MySQL, PostgreSQL or Oracle. For smaller setups and to get started, MySQL is very straight forward and will be described in here. For a production setup, PostgreSQL or Oracle can use partitioned table space that allows better data management over time. See https://github.com/ControlSystemStudio/phoebus/tree/master/app/databrowser-timescale for more on using PostgreSQL with the TimescaleDB extension to partition data and get optimized data retrieval.

Once the RDB is configured with the archive table schema, the archive engine is used both as a command line tool to configure the archive settings and as a service to write samples from PVs to the RDB. You can build the archive engine from sources or fetch a binary from https://controlssoftware.sns.ornl.gov/css_phoebus

### 9.5.1 Install MySQL (Centos Example)

Install:

```
sudo yum install mariadb-server
```

Start:

```
sudo systemctl start mariadb
```

Set root password, which is initially empty:

```
/usr/bin/mysql_secure_installation
```

In the following we assume you set the root password to `$root`. To start RDB when computer boots:

```
sudo systemctl enable mariadb.service
```

### 9.5.2 Create archive tables

Connect to mysql as root:

```
mysql -u root -p'$root'
```

and then paste the commands shown in `services/archive-engine/dbd/MySQL.dbd` (available online as https://github.com/ControlSystemStudio/phoebus/blob/master/services/archive-engine/dbd/MySQL.dbd ) to create the table setup for archiving PV samples.

The provided database schema is meant as an example, concentrating on the essential tables. It uses a single large `sample` table. A production setup might prefer to partition the table by for example creating a new partition each month.

The schema as provided does not rely on table constraints. For example, while the `chan_grp.eng_id` should refer to a valid `smpl_eng.eng_id`, there may not be a foreign key constraint to enforce this. This has been done to minimize RDB overhead, using the database simply as "storage" and enforcing the correctness of the data inside the archive engine when it is importing a configuration or adding samples. For a production setup, you may want to add or remove constraints as desired.

### 9.5.3 View Archive Data

The default settings for the Phoebus Data Browser check for archived data in `mysql://localhost/archive`. To access MySQL on another host, change these settings in your *Preferences Listing*

```
org.csstudio.trends.databrowser3/urls=jdbc:mysql://my.host.site.org/archive|RDB
org.csstudio.trends.databrowser3/archives=jdbc:mysql://my.host.site.org/archive|RDB
```

The `MySQL.dbd` used to install the archive tables adds a few demo samples for `sim://sine(0, 10, 50, 0.1)` around 2004-01-10 13:01, so you can simply add that channel to a Data Browser and find data at that time.

### 9.5.4 List, Export and Import Configurations

List configurations:

```
archive-engine.sh -list
Archive Engine Configurations:
ID  Name     Description        URL
 1  Demo     Demo Engine        http://localhost:4812
```

<<<<<<< Updated upstream Extract configuration into an XML file:: =======

Extract configuration into an XML file:: >>>>>>> Stashed changes

archive-engine.sh -engine Demo -export Demo.xml

For a description of the XML schema, see `archive_config.xsd`.

Modify the XML file or create a new one to list the channels you want to archive and to configure how they should be samples. For details on the 'scanned' and 'monitored' sample modes, refer to the CS-Studio manual chapter http://cs-studio.sourceforge.net/docbook/ch11.html

Finally, import the XML configuration into the RDB, in this example replacing the original one:

```
archive-engine.sh -engine Demo -import Demo.xml -port 4812 -replace_engine
```

### 9.5.5 PV Name Details

The archive engine uses CS-Studio PV names. "ca://xxxx" will force a Channel Access connection, "pva://xxxx" will force a PV Access connection, and just "xxxx" will use the default PV type configurable via

org.phoebus.pv/default=ca

Since EPICS 7, IOCs can support both protocols. "xxxx", "ca://xxxx" and "pva://xxxx" will thus refer to the same record on the IOC.

The preference setting

org.csstudio.archive/equivalent_pv_prefixes=ca, pva

causes the archive engine to treat them equivalent as well. For details, refer to the description of the *equivalent_pv_prefixes* preference setting.

### 9.5.6 Run the Archive Engine

To start the archive engine for a configuration:

```
archive-engine.sh -engine Demo -port 4812 -settings my_settings.ini
```

<<<<<<< Updated upstream

The engine name ('Demo') needs to match a previously imported configuration name, and the port number (4812) needs to match the port number used when importing the configuration. =======

The engine name ('Demo') needs to match a previously imported configuration name, and the port number (4812) needs to match the port number used when importing the configuration. >>>>>>> Stashed changes The settings (my_settings.ini) typically contain the EPICS CA address list settings as well as archive engine configuration details, see archive engine settings in *Preferences Listing*.

In a production setup, the archive engine is best run under `procServ` (https://github.com/ralphlange/procServ).

The running archive engine offers a simple shell:

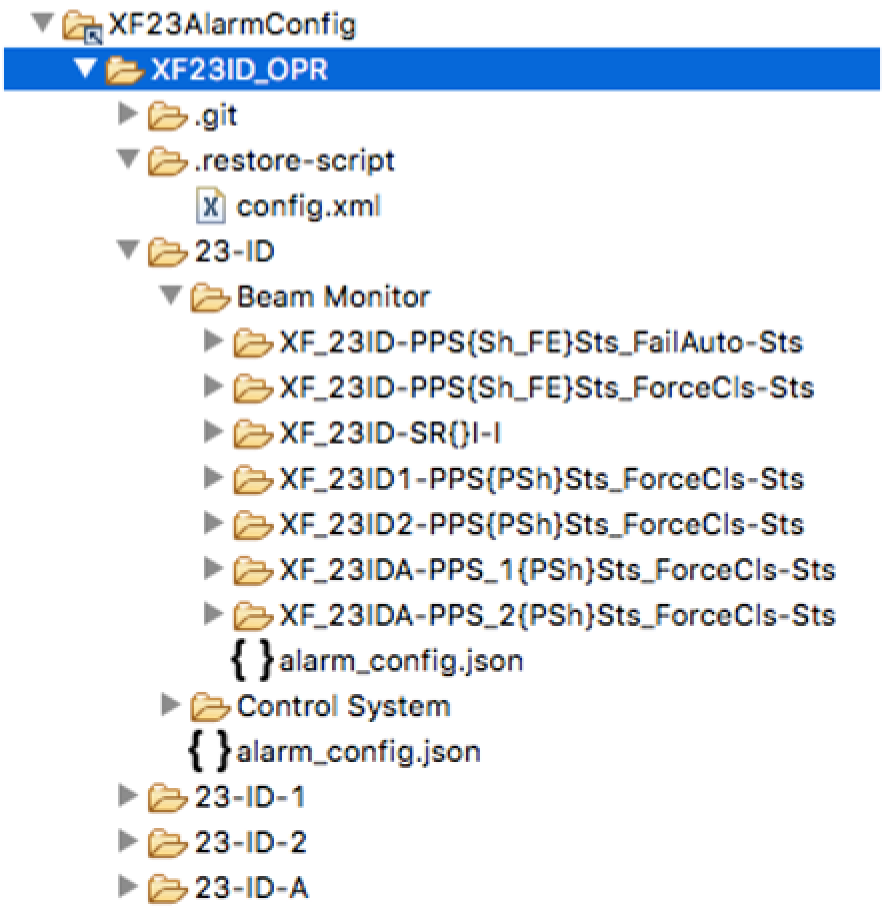
```
INFO Archive Configuration 'Demo'
...
INFO Web Server : http://localhost:4812
...
>
> help
Archive Engine Commands:
help             -  Show commands
disconnected     -  Show disconnected channels
restart          -  Restart archive engine
shutdown         -  Stop the archive engine
```

In addition, it has a web interface accessible under the URL shown at startup for inspecting connection state, last archived value for each channel and more. The engine can be shut down via either the `shutdown` command entered on the shell, or by accessing the `stop` URL. For the URL shown in the startup above that would be `http://localhost:4812/stop`.

## 9.6 Alarm Configuration Logging

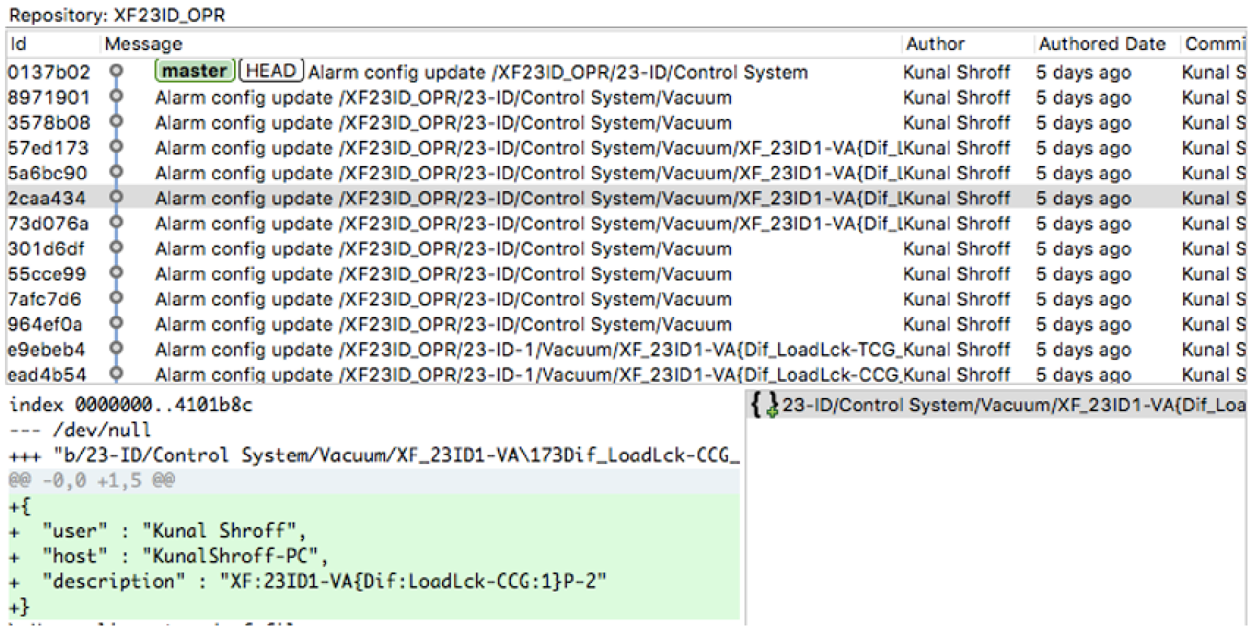
A simple service which logs all the configuration changes made to the alarm server configuration.

The alarm configuration tree is mapped to a directory structure where each node is represented by a directory and each leaf is a json file describing the alarm configuration for that element.



```
XF23AlarmConfig
  XF23ID_OPR
    .git
    .restore-script
      config.xml
    23-ID
      Beam Monitor
        XF_23ID-PPS{Sh_FE}Sts_FailAuto-Sts
        XF_23ID-PPS{Sh_FE}Sts_ForceCls-Sts
        XF_23ID-SR{}I-I
        XF_23ID1-PPS{PSh}Sts_ForceCls-Sts
        XF_23ID2-PPS{PSh}Sts_ForceCls-Sts
        XF_23IDA-PPS_1{PSh}Sts_ForceCls-Sts
        XF_23IDA-PPS_2{PSh}Sts_ForceCls-Sts
        alarm_config.json
      Control System
      alarm_config.json
    23-ID-1
    23-ID-2
    23-ID-A
```

The above file structure also uses the git version control system which allows us to trace all changes made to the alarm configuration.



Repository: XF23ID_OPR

| Id | Message | Author | Authored Date | Commi |
|---|---|---|---|---|
| 0137b02 | master HEAD Alarm config update /XF23ID_OPR/23-ID/Control System | Kunal Shroff | 5 days ago | Kunal S |
| 8971901 | Alarm config update /XF23ID_OPR/23-ID/Control System/Vacuum | Kunal Shroff | 5 days ago | Kunal S |
| 3578b08 | Alarm config update /XF23ID_OPR/23-ID/Control System/Vacuum | Kunal Shroff | 5 days ago | Kunal S |
| 57ed173 | Alarm config update /XF23ID_OPR/23-ID/Control System/Vacuum/XF_23ID1-VA{Dif_l | Kunal Shroff | 5 days ago | Kunal S |
| 5a6bc90 | Alarm config update /XF23ID_OPR/23-ID/Control System/Vacuum/XF_23ID1-VA{Dif_l | Kunal Shroff | 5 days ago | Kunal S |
| 2caa434 | Alarm config update /XF23ID_OPR/23-ID/Control System/Vacuum/XF_23ID1-VA{Dif_l | Kunal Shroff | 5 days ago | Kunal S |
| 73d076a | Alarm config update /XF23ID_OPR/23-ID/Control System/Vacuum/XF_23ID1-VA{Dif_l | Kunal Shroff | 5 days ago | Kunal S |
| 301d6df | Alarm config update /XF23ID_OPR/23-ID/Control System/Vacuum | Kunal Shroff | 5 days ago | Kunal S |
| 55cce99 | Alarm config update /XF23ID_OPR/23-ID/Control System/Vacuum | Kunal Shroff | 5 days ago | Kunal S |
| 7afc7d6 | Alarm config update /XF23ID_OPR/23-ID/Control System/Vacuum | Kunal Shroff | 5 days ago | Kunal S |
| 964ef0a | Alarm config update /XF23ID_OPR/23-ID/Control System/Vacuum | Kunal Shroff | 5 days ago | Kunal S |
| e9ebeb4 | Alarm config update /XF23ID_OPR/23-ID-1/Vacuum/XF_23ID1-VA{Dif_LoadLck-TCG_ | Kunal Shroff | 5 days ago | Kunal S |
| ead4b54 | Alarm config update /XF23ID_OPR/23-ID-1/Vacuum/XF_23ID1-VA{Dif_LoadLck-CCG | Kunal Shroff | 5 days ago | Kunal S |

```
index 0000000..4101b8c
--- /dev/null
+++ "b/23-ID/Control System/Vacuum/XF_23ID1-VA\173Dif_LoadLck-CCG_
@@ -0,0 +1,5 @@
+{
+  "user" : "Kunal Shroff",
+  "host" : "KunalShroff-PC",
+  "description" : "XF:23ID1-VA{Dif:LoadLck-CCG:1}P-2"
+}
```

23-ID/Control System/Vacuum/XF_23ID1-VA{Dif_Loa

Developer Documentation:

# CHAPTER 10

# Developer Information

To get started with developing for phoebus, refer to the instructions on https://github.com/ControlSystemStudio/phoebus .

CHAPTER 11

# Phoebus Product

The phoebus repo consists of a simple common product consisting of most of the commonly used applications.

The extensible and modular nature of the Phoebus framework allows for developers to assemble their own products. These products can be include or exclude applications and configurations to address the needs of a particular organization or workflow.

Detailed instructions along with some examples of site specific products can be found here

# CHAPTER 12

# Architecture

The fundamental phoebus architecture consists of **core** modules, user-interface related **core-ui** modules, and **app** modules. The core modules provide the infrastructure, while app modules implement specific application functionality. Everything is based on Java version 9 or higher, using Java FX as a graphics library.

A Phoebus product may contain just one application, for example only one of Probe, PV Tree, PV Table, Display Builder Runtime, so you end up with several Phoebus products that each perform one function. Alternatively, you can assemble a Phoebus product that contains all these applications. This allows integration between the applications, for example via context menus that start other PV-related applications based on the current selection.

## 12.1 Core Modules

**core-framework:** Fundamentals that many applications use, for example preferences, persistence, jobs, macros, localization, autocompletion.

Defines the `AppDescriptor` and `AppResourceDescriptor` Java Service Provider Interfaces (SPI) which are used to locate applications. Each application feature identifies itself by implementing an application description that describes to the Phoebus framework what the name of the application is, which types of resources (e.g. data files) it might accept, and most importantly how to start one or more instances of the application.

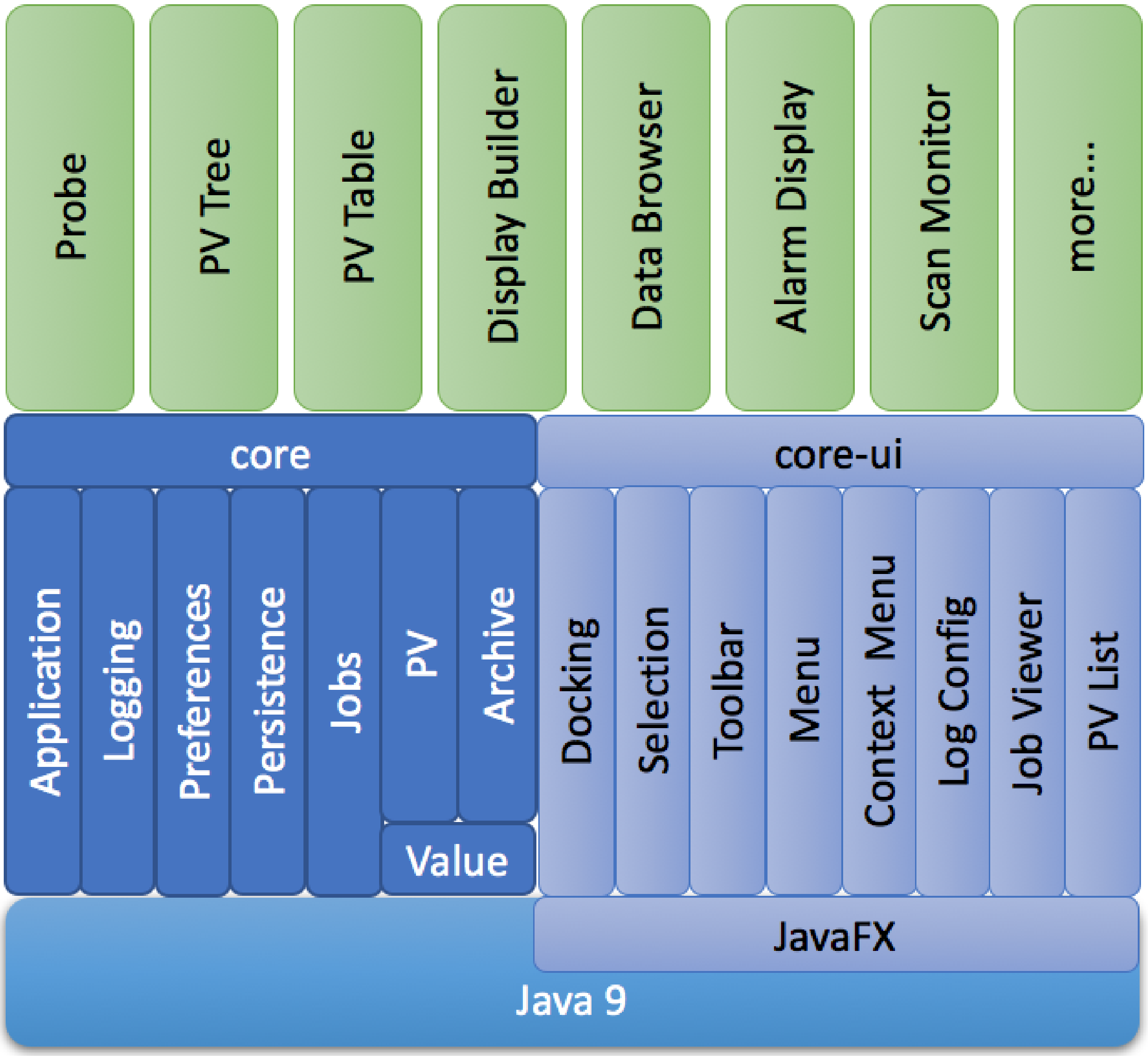
To create an `AppInstance`, i.e. an application instance, the framework invokes the `create()` method of the application descriptor. This will typically result in a new application instance, i.e. a new tab in the UI. Certain applications like the job viewer will create a singleton application instance.

On shutdown, the state of all windows and tabs is persisted in a memento file, and each `AppDescriptor` can also persist its own state. On startup, each window and tab is restored, the applications are restarted, and each application can restore its specific state from the memento.

The `JobManager` API allows submitting jobs based on a `JobRunnable` that supports progress reporting and cancellation.

**core-pv:** API for access to life data from Process Variables.

**core-logbook:** API for accessing a logbook, with SPI for site-specific implementations.


Probe
PV Tree
PV Table
Display Builder
Data Browser
Alarm Display
Scan Monitor
more...
core
core-ui
Application
Logging
Preferences
Persistence
Jobs
PV
Archive
Value
Docking
Selection
Toolbar
Menu
Context Menu
Log Config
Job Viewer
PV List
JavaFX
Java 9

**core-email:** API for creating emails.

**core-security:** API for authorization and secure storage.

**core-ui:** The `docking` package supports a window environment similar to a web browser. Each window can have multiple tabs. Users can move tabs between existing windows, or detach them into newly created windows.

The top-level Java FX `Node` for each application's UI scene graph is basically a `Tab`, wrapped in a Phoebus `DockItem` that tracks the `AppInstance` to allow it to be saved and restored.

The toolbar, main menu and context menus accept SPI-based contributions.

The `selection` package allows publishing and monitoring a selection of for example PVs.

The `undo` package simplifies the implementation of undo/redo functionality.

## 12.2 App Modules

Each app module implements a certain application feature, for example Probe, PV Tree, Display Builder. While application modules depend on one or more core modules, there should be few if no inter-application dependencies, allowing a product to contain only the desired applications.

## 12.3 Products

Each core and application module is built into a `jar` file. A product contains a `lib/` directory with the desired modules. When invoking the `Launcher`, it locates the available applications, menu and toolbar entries via SPI. Adding or removing Probe, PV Tree, .. from a product is done by simply adding or removing the respective jar file.

## 12.4 Services

Similar to a product, a service is a runnable tool, but typically for a non-UI functionality like a scan server, archive engine, alarm handler or logger.

# CHAPTER 13

# Locations

Phoebus uses the following Java System properties to locate help files, saved state etc.

These system variables are typically set automatically as described below, but when necessary they can be set when starting the product.

**`phoebus.install`:** Location where phoebus is installed. Has subdirectories `lib/`, `doc/`, and is used to locate the online help. Is automatically derived from the location of the framework JAR file.

**`phoebus.user`:** Location where phoebus keeps the memento and preferences. Defaults to `.phoebus` in the user's home directory.

## 13.1 Site-Specific Branding and Settings

The `phoebus.install` location is used for branding and site-specific settings.

**`settings.ini`:** At startup, Phoebus load preferences from this file if it is found in the install location. This allows packing site-specific settings into your product.

**`site_splash.png`:** This image will replace the default splash screen background with a site-specific version. It should be sized 480x300.

**`site_logo.png`:** This 64x64 sized image will replace the default window logo.

# CHAPTER 14

# Help System

Help files are **.rst* files with reStructuredText markup as described at http://www.sphinx-doc.org/en/stable/rest.html.

The top-level repository for help files is phoebus-doc (https://github.com/kasemir/phoebus-doc) and a snapshotData of the current version is accessible on http://phoebus-doc.readthedocs.io.

The top-level help repository provides the overall structure and content that describes the Phoebus-based CS-Studio in general. Each phoebus application can contribute help content that describes the specific application in more detail. This is done by adding a `doc/` folder with an `index.rst` file to the application sources. When phoebus-doc is built, it includes all `phoebus/**/doc/index.rst` in the Applications section of the manual. While the `*.rst` markup is ultimately converted into HTML, some applications might have already have HTML content generated by other means, for example from Javadoc. Any `doc/html` folder found in the applications source code is copied into the file html folder. To appear in the manual, it needs to be linked from the `index.rst` of an application via `raw` tags. For an example, refer to the display builder editor help.

Overall, the help content is thus generated from a combination of

1. Top-level content defined in `phoebus-doc/source/*.rst`
2. Application specific help copied from the `phoebus/**/doc/index.rst` source tree
3. Pre-generated HTML folders copied from the `phoebus/**/doc/html` source tree

In addition to building the help files locally as described below, or viewing a snapshotData of the current version online under the link mentioned above, the help content is also bundled into the phoebus-based CS-Studio product. When the phoebus product is built, it checks for the HTML version of the manual in `phoebus-doc/build/html`. If found, it is bundled into the product.

Complete build steps of manual and product:

```
# Obtain sources for documentation and product
git clone https://github.com/kasemir/phoebus-doc.git
git clone https://github.com/shroffk/phoebus.git

# Some application code may contain html content
# that needs to be generated, for example from Javadoc
( cd phoebus/app/display/editor;  ant -f javadoc.xml clean all )
```

(continues on next page)

(continued from previous page)

```
# Building the manual will locate and include
# all ../phoebus/applications/**/doc/index.rst
( cd phoebus-doc; make clean html )
# Windows: Use make.bat html

# Fetch dependencies
( cd phoebus/dependencies; mvn clean install )

# Build the product, which bundles help from
# ../phoebus-doc/build/html
# as phoebus-product/target/doc
( cd phoebus; ant clean dist )

# Could now run the product
( cd phoebus/phoebus-product; sh phoebus.sh )

# or distribute the ZIP file,
# phoebus/phoebus-product/target/phoebus-0.0.1.zip
```

## 14.1 Internals

In `phoebus-doc/source/conf.py`, the `createAppIndex()` method checks for the phoebus sources and builds the application section of the manual.

When invoking the Phoebus `Help` menu, it looks for a `doc/` folder in the installation location (see *Locations*).

As a fallback for development in the IDE, it looks for `phoebus-doc/build/html`.

# CHAPTER 15

# Localization

We encourage every developer to make their code localizable. The `NLS` class (based on the Eclipse RCP *NLS* idea) has been developed to aid in doing so.

Instead of hard-coding messages (UI strings), the `NLS` class allows us to put all a package's messages into a ResourceBundle called *messages*. This ResourceBundle is a collection of files called `messages.properties`, `messages_de.properties`, `messages_fr.properties`, etc. The `messages.properties` file contains the English localization, `messages_de.properties` contains the German localization, and so on. The appropriate messages (depending on the system locale and the `user.language` property) are loaded into the fields of a `Messages` class which we can then use in the UI code.

## 15.1 Creating a localization

To localize Phoebus to your language you need to find every `messages.properties` file in the project and create a translated version called `messages_xx.properties` where `xx` is the appropriate locale code (`fr`, `de`, `es`, etc.). You should also add your locale to the POM as described in the next section.

## 15.2 Checking the completeness of localizations

To ease the maintenance of localizations, a report can be generated to quickly find missing or extra messages compared to the default English localization.

In order to do so, you must first make sure that the locale you're interested in is listed in the `configuration/locales` section of the `l10n-status` Maven profile. In the following example, the report will include the German, Spanish and French localizations.

```
<!-- pom.xml (parent) -->
<project>
  ...
  <profiles>
    ...
```

(continues on next page)

(continued from previous page)

```
    <profile>
      <id>l10n-status</id>
      <reporting>
        <plugins>
          <plugin>
            <groupId>org.codehaus.mojo</groupId>
            <artifactId>l10n-maven-plugin</artifactId>
            <version>1.0-alpha-2</version>
            <configuration>
              <aggregate>true</aggregate>
              <includes>**/messages*.properties</includes>
              <locales>
                <locale>de</locale>
                <locale>es</locale>
                <locale>fr</locale>
              </locales>
            </configuration>
          </plugin>
          ...
        </plugins>
      </reporting>
    </profile>
      ...
  </profiles>
  ...
</project>
```

After that, you only need to run

```
mvn site -P l10n-status
```

The report will be located at `target/site/l10n-status.html`.

## 15.3 Writing localizable code

Suppose we want to be able to localize the following class:

```java
package org.phoebus.mypackage;

public class MyClass
{
    public void greet()
    {
        System.out.println("Hello");
        System.out.println("How are you today?");
    }
}
```

The first step is to create a `Messages.java` file with the following boilerplate:

```java
package org.phoebus.ui.mypackage;

import org.phoebus.framework.nls.NLS;

```

(continues on next page)

(continued from previous page)

```java
public class Messages
{
    public static String Hello;
    public static String HowAreYou;

    static
    {
        // initialize resource bundle
        NLS.initializeMessages(Messages.class);
    }

    private Messages()
    {
        // Prevent instantiation.
    }
}
```

Then, we replace the hard-coded strings in `MyClass` with `Messages`'s fields:

```java
package org.phoebus.mypackage;

import org.phoebus.mypackage.Messages;

public class MyClass
{
    public void greet()
    {
        System.out.println(Messages.Hello);
        System.out.println(Messages.HowAreYou);
    }
}
```

Finally, we create the *messages* ResourceBundle with all the localizations we want.

messages.properties:

```
Hello=Hello
HowAreYou=How are you doing today?
```

messages_es.properties:

```
Hello=Hola
HowAreYou=¿Cómo estás hoy?
```

# CHAPTER 16

# Preferences Listing

The following preference settings are available for the various application features. To use them in your settings file, remember to prefix each setting with the package name.

## 16.1 alarm

File ../../app/alarm/model/src/main/resources/alarm_preferences.properties:

```
# --------------------------------------
# Package org.phoebus.applications.alarm
# --------------------------------------

# Kafka Server host:port
server=localhost:9092

# A file to configure the properites of kafka clients
kafka_properties=

# Name of alarm tree root
config_name=Accelerator

# Names of selectable alarm configurations
# The `config_name` will be used as the default for newly opened tools,
# and if `config_names` is empty, it remains the only option.
# When one or more comma-separated configurations are listed,
# the UI shows the selected name and allows switching
# between them.
config_names=Accelerator, Demo

# Timeout in seconds for initial PV connection
connection_timeout=30

# Timeout in seconds for "sevrpv:" updates
```

(continues on next page)

(continued from previous page)

```
severity_pv_timeout=5

## Area Panel

# Item level for alarm area view:
# 1 - Root element
# 2 - Top-level "area" elements just below root
# 3 - Show all the items at level 3
alarm_area_level=2

# Number of columns in the alarm area view
alarm_area_column_count=3

# Gap between alarm area panel items
alarm_area_gap=5

# Font size for the alarm area view
alarm_area_font_size=15

# Limit for the number of context menu items.
# Separately applied to the number of 'guidance',
# 'display' and 'command' menu entries.
alarm_menu_max_items=10

# Initial Alarm Tree UI update delay [ms]
#
# The initial flurry of alarm tree updates can be slow
# to render. By allowing the alarm client to accumulate
# alarm tree information for a little time and then
# performing an initial bulk representation, the overall
# alarm tree startup can be faster, especially when
# the UI is viewed via a remote desktop
#
# Set to 0 for original implementation where
# all alarm tree items are added to the model
# as they are received in initial flurry of updates.
alarm_tree_startup_ms=2000

# Order of columns in alarm table
# Allows re-ordering as well as omitting columns
alarm_table_columns=Icon, PV, Description, Alarm Severity, Alarm Status, Alarm Time,␣
↪Alarm Value, PV Severity, PV Status

# By default, the alarm table uses the common alarm severity colors
# for both the text color and the background of cells in the "Severity" column.
#
# Older implementations always used the background to indicate alarm severity,
# and this options emulates that by using the alarm severity text(!) color
# for the background, automatically using black or white for the text
# based on brightness.
alarm_table_color_legacy_background=true

# Alarm table row limit
# If there are more rows, they're suppressed
alarm_table_max_rows=2500

# Directory used for executing commands
```

(continues on next page)

(continued from previous page)

```
# May use Java system properties like this: $(prop_name)
command_directory=$(user.home)

# The threshold of messages that must accumulate before the annunciator begins to␣
↪simply state: "There are X Alarm messages."
annunciator_threshold=3

# The number of messages the annunciator will retain before popping messages off the␣
↪front of the message queue.
annunciator_retention_count=100

# Timeout in seconds at which server sends idle state updates
# for the 'root' element if there's no real traffic.
# Client will wait 3 times this long and then declare a timeout.
idle_timeout=10

# Name of the sender, the 'from' field of automated email actions
automated_email_sender=Alarm Notifier <alarm_server@example.org>

# Comma-separated list of automated actions on which to follow up
# Options include mailto:, cmd:
automated_action_followup=mailto:, cmd:

# Optional heartbeat PV
# When defined, alarm server will set it to 1 every heartbeat_secs
#heartbeat_pv=Demo:AlarmServerHeartbeat
heartbeat_pv=

# Heartbeat PV period in seconds
heartbeat_secs=10

# Period for repeated annunciation
#
# If there are active alarms, i.e. alarms that have not been acknowleded,
# a message "There are 47 active alarms" will be issued
#
# Format is HH:MM:SS, for example 00:15:00 to nag every 15 minutes.
# Set to 0 to disable
nag_period=00:15:00

# Connection validation period in seconds
#
# Server will check the Kafka connection at this period.
# After re-establishing the connection, it will
# re-send the state of every alarm tree item.
# Set to 0 to disable.
connection_check_secs=5

# To turn on disable notifications feature, set the value to true
disable_notify_visible=false

# Options for the "Disable until.." shortcuts in the PV config dialog
#
# Comma separated, each option needs to comply with TimeParser.parseTemporalAmount():
# 30 seconds, 5 minutes, 1 hour, 6 hours, 1 day, 30 days, ...
shelving_options=1 hour, 6 hours, 12 hours, 1 day, 7 days, 30 days
```

(continues on next page)

(continued from previous page)

```
# Macros for UI display, command or web links
#
# Format: M1=Value1, M2=Value2
macros=TOP=/home/controls/displays,WEBROOT=http://localhost/controls/displays
```

## 16.2 alarm.logging.ui

File ../../app/alarm/logging-ui/src/main/resources/alarm_logging_preferences.properties:

```
# -------------------------------------------------
# Package org.phoebus.applications.alarm.logging.ui
# -------------------------------------------------

service_uri = http://localhost:9000
results_max_size = 10000
```

## 16.3 archive

File ../../services/archive-engine/src/main/resources/archive_preferences.properties:

```
# ----------------------------
# Package org.csstudio.archive
# ----------------------------

# RDB URL for archived data
#
# Oracle example
# url=jdbc:oracle:thin:user/password@//172.31.73.122:1521/prod
#
# PostgreSQL example
# url=jdbc:postgresql://localhost/archive
#
# MySQL example
url=jdbc:mysql://localhost/archive?rewriteBatchedStatements=true

# RDB user and password
# Some applications also provide command-line option to override.
user=archive
password=$archive

# Schema name. Used with an added "." as prefix for table names.
# For now this is only used with Oracle URLs and ignored for MySQL
schema=

# Timeout [seconds] for certain SQL queries
# Fundamentally, the SQL queries for data take as long as they take
# and any artificial timeout just breaks queries that would otherwise
# have returned OK  few seconds after the timeout.
# We've seen Oracle lockups, though, that caused JDBC to hang forever
# because the SAMPLE table was locked. No error/exception, just hanging.
# A timeout is used for operations other than getting the actual data,
```

(continues on next page)

(continued from previous page)

```
# for example the channel id-by-name query which _should_ return within
# a shot time, to catch that type of RDB lockup.
# timeout_secs=120
# With PostgreSQL, the setQueryTimeout API is not implemented,
# and calling it results in an exception.
# Setting the timeout to 0 disables calls to setQueryTimeout.
timeout_secs=0

# Use a blob to read/write array samples?
#
# The original SAMPLE table did not contain an ARRAY_VAL column
# for the array blob data, but instead used a separate ARRAY_VAL table.
# When running against an old database, this parameter must be set to false.
use_array_blob=true

# Name of sample table for writing
write_sample_table=sample

# Maximum length of text samples written to SAMPLE.STR_VAL
max_text_sample_length=80

# Use postgres copy instead of insert
use_postgres_copy=false

# Channel names use a prefix ca://, pva://, loc://, ...
# to select the type of PV or network protocol.
# The preference setting
#
#  org.phoebus.pv/default=ca
#
# determines the default type when no prefix is provided.
#
# With EPICS IOCs from release 7 on, the PVs
# "xxx", "ca://xxx" and "pva://xxx" all refer
# to the same record "xxx" on the IOC.
#
# The archive configuration stores the PV name as given.
# It is used as such when connecting to the live data source,
# resulting in "ca://.." or "pva://.." connections as requested.
# Samples are written to the archive under that channel name.
#
# This archive engine preference setting establishes one or more prefixes
# as equal when importing an engine configuration.
# For example, assume
#
#  equivalent_pv_prefixes=ca, pva
#
# When adding a PV "pva://xxx" to the configuration,
# we check if the archive already contains a channel "xxx", "ca://xxx" or "pva://xxx".
# If any of them are found, the `-import` will consider "pva://xxx" as a duplicate.
#
# When importing a PV "pva://xxx" into a sample engine configuration that already
# contains the channel "ca://xxx" or "xxx", the channel will be renamed,
# so that engine will from now on use "pva://xxx".
#
# When importing a PV "pva://xxx" into a configuration that already
# contains a different engine setup with the channel "ca://xxx" or "xxx",
```

(continues on next page)

(continued from previous page)

```
# the channel will by default rename unchanged, so "ca://xxx" or "xxx"
# will remain in their original engine setup, "pva://xxx" will be skipped.
#
# When using `-import` with the additional `-steal_channels` option,
# the existing "...xxx" channel will be renamed to "pva://xxx" and moved
# to the imported engine configuration.
#
# When `equivalent_pv_prefixes` is empty,
# any PV name is used as is without looking for equivalent names.
# So "xxx", "ca://xxx" and "pva://xxx" can then all be imported
# as separate channels, which is likely wrong because it would simply
# store data from the same underlying record more than once.
#
# This default should be the most practical setting when adding
# EPICS 7 IOCs and starting to transition towards "pva://..".
# Existing "xxx" or "ca://xxx" channels can thus be renamed
# to "pva://xxx" while retaining their sample history.
#
# Note that the data browser has a similar `equivalent_pv_prefixes`
# setting to search for a channel name in several variants.
equivalent_pv_prefixes=ca, pva

# Seconds between log messages for Not-a-Number, futuristic, back-in-time values,␣
↪buffer overruns
# 24h = 24*60*60 = 86400
log_trouble_samples=86400
log_overrun=86400

# Write period in seconds
write_period=30

# Maximum number of repeat counts for scanned channels
max_repeats=60

# Write batch size
batch_size=500

# Buffer reserve (N times what's ideally needed)
buffer_reserve=2.0

# Samples with time stamps this far ahead of the local time
# are ignored
# 24*60*60 = 86400 = 1 day
ignored_future=86400
```

## 16.4 archive.reader.appliance

File ../../app/databrowser/src/main/resources/appliance_preferences.properties:

```
# ----------------------------------------
# Package org.phoebus.archive.reader.appliance
# ----------------------------------------

useStatisticsForOptimizedData=true
```

(continues on next page)

(continued from previous page)

```
useNewOptimizedOperator=true

# Use 'https://..' instead of plain 'http://..' ?
useHttps=false
```

## 16.5 archive.reader.channelarchiver

File ../../app/databrowser/src/main/resources/channelarchiver_preferences.properties:

```
# --------------------------------------------------
# Package org.phoebus.archive.reader.channelarchiver
# --------------------------------------------------

# Use 'https://..' instead of plain 'http://..' ?
use_https=false
```

## 16.6 archive.reader.rdb

File ../../app/databrowser/src/main/resources/archive_reader_rdb_preferences.properties:

```
---------------------------------------
# Package org.phoebus.archive.reader.rdb
# --------------------------------------

# User and password for reading archived data
user=archive
password=$archive

# Table prefix
# For Oracle, this is typically the schema name,
# including "."
prefix=

# Timeout [seconds] for certain SQL queries
# Fundamentally, the SQL queries for data take as long as they take
# and any artificial timeout just breaks queries that would otherwise
# have returned OK a few seconds after the timeout.
# We've seen Oracle lockups, though, that caused JDBC to hang forever
# because the SAMPLE table was locked. No error/exception, just hanging.
# A timeout is used for operations other than getting the actual data,
# for example the channel id-by-name query which _should_ return within
# a shot time, to catch that type of RDB lockup.
timeout_secs=120
# Setting the timeout to 0 disables calls to setQueryTimeout,
# which may be required for PostgreSQL where the setQueryTimeout API is not␣
↪implemented.
# timeout_secs=0


# Use a BLOB to read array samples?
#
```

(continues on next page)

(continued from previous page)

```
# The original SAMPLE table did not contain an ARRAY_VAL column
# for the array blob data, but instead used a separate ARRAY_VAL table.
# When running against an old database, this parameter must be set to false.
use_array_blob=true

# Use stored procedures and functions for 'optimized' data readout?
# Set to procedure name, or nothing to disable stored procedure.
stored_procedure=
starttime_function=

# MySQL:
# stored_procedure=archive.get_browser_data

# PostgreSQL
# stored_procedure=public.get_browser_data

# Oracle:
# stored_procedure=chan_arch.archive_reader_pkg.get_browser_data
# starttime_function=SELECT chan_arch.archive_reader_pkg.get_actual_start_time (?, ?,␣
↪?)  FROM DUAL


# JDBC Statement 'fetch size':
# Number of samples to read in one network transfer.
#
# For Oracle, the default is 10.
# Tests resulted in a speed increase up to fetch sizes of 1000.
# On the other hand, bigger numbers can result in java.lang.OutOfMemoryError.
fetch_size=1000
```

## 16.7 archive.ts

File ../../app/databrowser-timescale/src/main/resources/archive_ts_preferences.properties:

```
--------------------------------
# Package org.csstudio.archive.ts
# -------------------------------

# User and password for reading archived data
user=report
password=$report

# Timeout [seconds] for certain SQL queries, 0 to disable timeout.
# Fundamentally, the SQL queries for data take as long as they take
# and any artificial timeout just breaks queries that would otherwise
# have returned OK a few seconds after the timeout.
# A timeout is used for operations other than getting the actual data,
# for example the channel id-by-name query which _should_ return within
# a short time.
timeout_secs=120

# JDBC Statement 'fetch size':
# Number of samples to read in one network transfer.
# Speed tends to increase with fetch size.
```

(continues on next page)

(continued from previous page)

```
# On the other hand, bigger numbers can result in java.lang.OutOfMemoryError.
fetch_size=10000
```

## 16.8 channel.views.ui

File ../../app/channel/views/src/main/resources/cv_preferences.properties:

```
# --------------------------------------
# Package org.phoebus.channel.views.ui
# --------------------------------------

# Show the active PVs only
show_active_cb=false
```

## 16.9 channelfinder

File ../../app/channel/channelfinder/src/main/resources/channelfinder_preferences.properties:

```
# ----------------------------------------
# Package org.phoebus.channelfinder
# ----------------------------------------

serviceURL=http://localhost:8080/ChannelFinder
username=admin
password=adminPass

rawFiltering=false
```

## 16.10 console

File ../../app/console/src/main/resources/console_preferences.properties:

```
# ----------------------------------------
# Package org.phoebus.applications.console
# ----------------------------------------

# Number of output lines to keep.
# Older output is dropped.
output_line_limit=100

# Number of lines to keep in input history,
# accessible via up/down cursor keys
history_size=20

# Font name and size
font_name=Liberation Mono
font_size=14

# Prompt (may include trailing space)
```

(continues on next page)

(continued from previous page)

```
prompt=>>>\

# Prompt (input field) info
prompt_info=Enter console command

# 'Shell' to execute.
#
# Examples:
#   /usr/bin/python -i
#   /usr/bin/python -i /path/to/some/initial_file.py
#   /bin/bash
#
# Value may include properties.
shell=/usr/bin/python -i

# Folder where the shell process should be started
#
# Value may include properties.
directory=$(user.home)
```

## 16.11 display.builder.editor

File ../../app/display/editor/src/main/resources/display_editor_preferences.properties:

```
# ----------------------------------------
# Package org.csstudio.display.builder.editor
# ----------------------------------------

# Widget types to hide from the palette
#
# Comma separated list of widget types that will not be shown
# in the palette.
# Existing displays that use these widgets can still be edited
# and executed, but widgets do not appear in the palette to
# discourage adding them to new displays.

# Hiding widgets where representation has not been imported because of dependencies
hidden_widget_types=linear-meter,knob,gauge,clock,digital_clock
#
#
# GUI Menu action Applications / Display / New Display opens the following template
new_display_template=examples:/initial.bob

# Size of undo stack. Defaults to 50 if not set.
undo_stack_size=50
```

## 16.12 display.builder.model

File ../../app/display/model/src/main/resources/display_model_preferences.properties:

```
# ----------------------------------------
# Package org.csstudio.display.builder.model
# ----------------------------------------


# Widget classes
# One or more *.bcf files, separated by ';'
# Defaults to built-in copy of examples/classes.bcf
class_files=examples:classes.bcf

# Named colors
# One or more *.def files, separated by ';'
# Defaults to built-in copy of examples/color.def
color_files=examples:color.def

# Named fonts
# One or more *.def files, separated by ';'
# Defaults to built-in copy of examples/font.def
font_files=examples:font.def

# Global macros, used for all displays.
#
# Displays start with these macros,
# and can then add new macros or overwrite
# the values of these macros.
#
# Format:
# Entries where the XML tag name is the macro name,
# and the XML content is the macro value.
# The macro name must be a valid XML tag name:
# * Must start with character
# * May then contain characters or numbers
# * May also contain underscores
#
macros=<EXAMPLE_MACRO>Value from Preferences</EXAMPLE_MACRO><TEST>true</TEST>


# Timeout [ms] for loading files: Displays, but also color, font, widget class files
read_timeout=10000

# Timeout [sec] for caching files loaded from a URL
cache_timeout=60


# 'BOY' *.opi files provide the font size in 'points'.
# All other positions and sizes are in 'pixels'.
# A point is meant to represent 1/72th of an inch.
# The actual on-screen size display settings.
# Plugging a different monitor into the computer can
# potentially change the DPI settings of the graphics driver,
# resulting in different font sizes.
# The display builder uses fonts in pixels to avoid such changes.
#
# When reading legacy display files, we do not know the DPI
# scaling that was used to create the display.
# This factor is used to translate legacy font sizes
# from 'points' into 'pixel':
```

(continues on next page)

(continued from previous page)

```
#
# legacy_points = pixel * legacy_font_calibration
#
# The test program
#   org.csstudio.display.builder.representation.swt.SWTFontCalibation
# can be used to obtain the factor when executed on the original
# platform where the legacy display files were created.
#
# When loading legacy files,
# _increasing_ the legacy_font_calibration will
# result in _smaller_ fonts in the display builder
legacy_font_calibration=1.01

# Maximum re-parse operations
#
# When reading legacy *.opi files and for example
# finding a "TextUpdate" widget that has no <pv_name>,
# it will be changed into a "Label" widget and then re-parsed.
# If more than a certain number of re-parse operations are triggered
# within one 'level' of the file (number of widgets at the root of the display,
# or number of childred for a "Group" widget),
# the parser assumes that it entered an infinite re-parse loop
# and aborts.
max_reparse_iterations=5000

# Create display file with comments?
with_comments=false

# When writing a display file, skip properties that are still at default values?
skip_defaults=true
```

## 16.13 display.builder.representation

File ../../app/display/representation/src/main/resources/display_representation_preferences.properties:

```
# ---------------------------------------------------
# Package org.csstudio.display.builder.representation
# ---------------------------------------------------

## Representation Tuning
#
# The representation 'throttles' updates to widgets.
# When a widget requests an update, a little accumulation time
# allows more updates to accumulate before actually performing
# the queued update requests on the UI thread.
#
# An update delay then suppresses further updates to prevent
# flooding the UI thread.
#
# Update runs that last longer than a threshold can be logged

# Time waited after a trigger to allow for more updates to accumulate
update_accumulation_time = 20

```

(continues on next page)

(continued from previous page)

```
# Pause between updates to prevent flooding the UI thread
update_delay = 100

# Period in seconds for logging update performance
performance_log_period_secs = 5

# UI thread durations above this threshold are logged
performance_log_threshold_ms = 20

# Pause between updates of plots (XY, lines)
# Limit to 250ms=4 Hz
plot_update_delay = 250

# Pause between updates of image plots
# Limit to 250ms=4 Hz
image_update_delay = 250

# Length limit for tool tips
# Tool tips that are too long can be a problem
# on some window systems.
tooltip_length=150

# Timeout for load / unload of Embedded Widget content [ms]
embedded_timeout=5000
```

## 16.14 display.builder.representation.javafx

File ../../app/display/representation-javafx/src/main/resources/jfx_repr_preferences.properties:

```
# ----------------------------------------------------------
# Package org.csstudio.display.builder.representation.javafx
# ----------------------------------------------------------

# When clicking on the 'slider' widget 'track',
# should the value increment/decrement,
# matching the behavior of EDM, BOY, ...?
# Otherwise, jump to the clicked value right away.
inc_dec_slider=true

# How does mouse need to hover until tool tip appears?
tooltip_delay_ms=250

# Once displayed, how long does the tool tip remain visible?
tooltip_display_sec=30

# Note that for historic reasons tool tips are also influenced
# by the property `org.csstudio.display.builder.disable_tooltips`.
# When `true`, tool tips are disabled.
```

## 16.15 display.builder.runtime

File ../../app/display/runtime/src/main/resources/display_runtime_preferences.properties:

```
# ----------------------------------------------
# Package org.csstudio.display.builder.runtime
# ----------------------------------------------

# Search path for Jython scripts used by the display runtime.
# Note that format depends on the OS.
# On UNIX systems, path entries are separated by ':', on Windows by ';'.
# python_path=/home/controls/displays/scripts:/home/fred/my_scripts
python_path=

# PV Name Patches
#
# Translate PV names based on regular expression pattern and replacement
#
# Format:  pattern@replacement@pattern@replacement
#
# Setting must contain a sequence of pattern & replacement pairs,
# all separated by '@'.
#
# The regular expression for the pattern can includes "( )" groups,
# which are then used in the replacement via "$1", "$2", ..
#
# If the item separator character '@' itself is required within the pattern or␣
↪replacement,
# use '[@]' to distinguish it from the item separator, i.e.
#
#    [@]work@[@]home
#
# will patch "be@work" -> "be@home"
#
# Patches are applied in the order they're listed in the preference, i.e.
# later patches are applied to names already patched by earlier ones.
#
# Example:
# Remove PVManager's longString modifier,              'some_pv {"longString":true}' ->
↪ 'some_pv'
# turn constant formula into constant local variable, '=42'                          ->
↪ 'loc://const42(42)'
# as well as constant name into constant local var,   '="Fred"'                      ->
↪ 'loc://strFred("Fred")'
pv_name_patches=\\{"longString":true\\}"@@^="([a-zA-Z]+)"@loc://str$1("$1")

# PV update throttle in millisecs
# 250ms = 4 Hz
update_throttle=250

# "Probe Display"
# Added to context menu for ProcessVariables,
# invoked with macro PV set to the PV name.
# When left empty, the "Probe Display"
# context menu entry is disabled.
probe_display=examples:/probe.bob
```

## 16.16 display.converter.edm

File ../../app/display/convert-edm/src/main/resources/edm_converter_preferences.properties:

```
# ------------------------------------------
# Package org.csstudio.display.converter.edm
# ------------------------------------------

# Path to the directory where the auto-converter will
# generate auto-converted files.
# May include system properties like $(user.home).
# Target directory must be in the file system.
# The folder is created if it doesn't exist.
#
# When left empty, the auto-converter is disabled.
auto_converter_dir=

# Path (prefix) that will be stripped from the original
# EDM file name before converting.
# When empty, the complete path will be stripped.
#
# For example, assume we need to convert
#  /path/to/original/vacuum/segment1/vac1.edl
#
# With an empty auto_converter_strip,
# this will be converted into {auto_converter_dir}/vac1.edl
#
# With auto_converter_strip=/path/to/original,
# it will be converted into {auto_converter_dir}/vacuum/segment1/vac1.edl
auto_converter_strip=

# EDM colors.list file
# Must be defined to use converter.
# May be a file system path or http:/.. link
colors_list=

# Font mappings
#
# Format: EDMFontPattern=DisplayBuilderFont,Pattern=Font,...
# EDMFontPattern is regular expression for the name used by EDM
#
# Patterns are checked in the order in which they're listed in here,
# so a catch-all ".*" pattern should be at the end
font_mappings=helvetica=Liberation Sans,courier=Liberation Mono,times=Liberation␣
↪Serif,.*=Liberation Sans

# Path to text file that lists EDM search paths.
# May be a file system path or http:/.. link.
#
# In the file, each line in the text file contains a path,
# which may be a file system path or a http:// link.
# When trying to open an *.edl file,
# converter will try each path in the order
# listed in the file.
# Lines starting with "#" are ignored.
#
# When the edm_paths_config is left empty,
# the converter won't find files.
edm_paths_config=

# Pattern and replacement for patching paths to *.stp (StripTool) files
```

(continues on next page)

(continued from previous page)

```
#
# 'Shell Command' buttons in EDM that invoke a command of the form
#
#     StripTool /some/path/to/plot.stp
#
# are converted into ActionButtons which open the `/some/path/to/plot.stp` file.
# Data Browser will then open the file when the action is invoked.
#
# The following regular expression pattern and replacement can be used
# to patch `/some/path/to/plot.stp`.
# By default, both are empty, so the path remains unchanged.
#
# Example for transforming all absolute paths into a web location:
#
# stp_path_patch_pattern=^(/)
# stp_path_patch_replacement=https://my_web_server/stripcharts$1
#
# Note how the pattern may include group markers (..)
# and the replacement can reference them via $1, $2, ...
stp_path_patch_pattern=
stp_path_patch_replacement=
```

## 16.17 email

File ../../core/email/src/main/resources/email_preferences.properties:

```
# -------------------------
# Package org.phoebus.email
# -------------------------

# smtp host
# When set to "DISABLE", email support is disabled
mailhost=smtp.bnl.gov

# smtp port
mailport=25

# User and password for connecting to the mail host, usually left empty
username=
password=

# Default address to be used for From:
# if it is left empty then the last used from address is used
from=
```

## 16.18 errlog

File ../../app/errlog/src/main/resources/errlog_preferences.properties:

```
# ---------------------------------------
# Package org.phoebus.applications.errlog
```

(continues on next page)

(continued from previous page)

```
# --------------------------------------

# Number of lines to keep in error log
max_lines = 500
```

## 16.19 eslog

File ../../app/eslog/src/main/resources/eslog_preferences.properties:

```
# -------------------------------------
# Package org.phoebus.applications.eslog
# -------------------------------------
es_url=
es_index=messagelog

jms_url=
jms_user
jms_password
jms_topic=LOG
```

## 16.20 filebrowser

File ../../app/filebrowser/src/main/resources/filebrowser_preferences.properties:

```
# -------------------------------------------
# Package org.phoebus.applications.filebrowser
# -------------------------------------------

# Initial root directory for newly opened file browser
# May use system properties like "$(user.home)".
# At runtime, user can select a different base directory,
# but pressing the "Home" button reverts to this one.
default_root=$(user.home)

# Show hidden files (File.isHidden)?
show_hidden=false
```

## 16.21 framework.autocomplete

File ../../core/framework/src/main/resources/autocomplete_preferences.properties:

```
# -----------------------------------------
# Package org.phoebus.framework.autocomplete
# -----------------------------------------

# Enable the built-in PV proposal providers?
enable_loc_pv_proposals=true
enable_sim_pv_proposals=true
enable_sys_pv_proposals=true
```

(continues on next page)

(continued from previous page)

```
enable_pva_pv_proposals=true
enable_mqtt_pv_proposals=false
enable_formula_proposals=true

# Site-specific proposal providers can be added via PVProposalProvider SPI,
# and disabled by removing the contribution.
```

## 16.22 framework.workbench

File ../../core/framework/src/main/resources/workbench_preferences.properties:

```
# ---------------------------------------
# Package org.phoebus.framework.workbench
# ---------------------------------------

# External applications
#
# Defines applications to use for specific file extensions
#
# Format:
#
# Each definition consists of name, file extensions, command.
#
# Name is the name of the definition, used to register the application.
# File extensions is a '|'-separated list of file extensions (not including the 'dot
↪').
# Command is the path to the command.
# The command will be invoked with the full path to the resource as an argument.
#
# Each definition must use a key that starts with "external_app_"

# Examples:
#
# Start 'gedit' for text files
# external_app_text=Text Editor,
↪txt|dat|py|ini|db|xml|xsl|css|cmd|sh|st|log|out|md|shp,gedit
#
# Start 'eog' for images, 'firefox' for PDF files
# external_app_image=Image Viewer,png|jpg|gif|jpeg,eog
#
# Start 'firefox' to view PDFs
# external_app_pdf=PDF Viewer,pdf,firefox
#
# Example for some site-specific tool that opens 'alog' files
# external_app_alog=Alignment Log,alog,/path/to/alog_viewer

# Directory where external applications are started
# May use system properties
external_apps_directory=$(user.home)
```

## 16.23 imageviewer

File ../../app/imageviewer/src/main/resources/image_viewer_preferences.properties:

```
# --------------------------------------------
# Package org.phoebus.applications.imageviewer
# --------------------------------------------

# Watermark text
watermark_text=W A T E R M A R K
```

## 16.24 javafx.rtplot

File ../../app/rtplot/src/main/resources/rt_plot_preferences.properties:

```
# ----------------------------------
# Package org.csstudio.javafx.rtplot
# ----------------------------------

# Coloring used to shade plot region beyond 'now'
# in time-based plots. RGBA (all values 0..255)
# Painted on on top of grid, before traces are drawn.
#
# Half-transparent, average of black & white,
# works for both white and black backgrounds
shady_future=128, 128, 128, 128

# If you prefer a rose-colored future
# shady_future=255, 128, 128, 25

# If you prefer to not highlight the plot region beyond 'now'
# shady_future=128, 128, 128, 0
```

## 16.25 logbook

File ../../core/logbook/src/main/resources/logbook_preferences.properties:

```
# ------------------------------
# Package org.phoebus.logbook
# ------------------------------

# Site specific log book client implementation name.
# When empty, logbook submissions are disabled
logbook_factory=inmemory

# Determines if a log entry created from context menu (e.g. display or data browser)
# should auto generate a title (e.g. "Display Screenshot...").
auto_title=true

# Determines if a log entry created from context menu (e.g. display or data browser)
# should auto generate properties (e.g. "resources.file").
auto_property=false
```

## 16.26 logbook.olog.ui

File ../../app/logbook/olog/ui/src/main/resources/log_olog_ui_preferences.properties:

```
# ------------------------------
# Package org.phoebus.logbook.olog.ui
# ------------------------------

# Comma-separated list of default logbooks for new log entries.
default_logbooks=Scratch Pad

# The default query for logbook applications
default_logbook_query=desc=*&start=12 hours&end=now

# Whether or not to save user credentials to file so they only have to be entered␣
↪once when making log entries.
save_credentials=false

# Stylesheet for the items in the log calendar view
calendar_view_item_stylesheet=Agenda.css

# Text to render for the "Level" field of a log entry. Sites may wish to customize␣
↪this with respect to
# its wording and its implied purpose.
level_field_name=Level:

# Name of markup help. Language resolution and file extension is handled on service.
markup_help=CommonmarkCheatsheet

# Root URL of the Olog web client, if one exists. Set this to the empty string
# to suppress rendering of the "Copy URL" button for a log entry.
web_client_root_URL=

# Log entry groups support. If set to false user will not be able to create replies
# to log entries, and consequently UI elements and views related to log entry
# groups will not be shown.
log_entry_groups_support=false

# Comma separated list of "hidden" properties. For instance, properties that serve␣
↪internal
# business logic, but should not be rendered in the properties view.
hidden_properties=Log Entry Group

# Log Entry Table display name. If non-empty it overrides default "Log Entry Table"
log_entry_table_display_name=

# Log Entry Calendar display name. If non-empty it overrides default "Log Entry␣
↪Calendar"
log_entry_calendar_display_name=

# Log Entry property attribute types.
# The preference should be a URL pointing to an attribute_type.properties file.
# e.g. log_attribute_desc=file:///C:/phoebus/app/logbook/olog/ui/src/main/resources/
↪org/phoebus/logbook/olog/ui/log_property_attributes.properties
# Classpath resource is supported if specified like log_attribute_desc=classpath:my_
↪attr.properties. In this
# example the my_attr.properties file must be bundled as a classpath resource in the␣
↪package org.phoebus.logbook.olog.ui.
```

(continues on next page)

(continued from previous page)

```
# This optional file describing special types associated with some property␣
↪attributes.
#
log_attribute_desc=

# Limit used in "paginated" search, i.e. the number of search results per page
search_result_page_size=30

# Number of queries maintained by the OlogQueryManager. To make sense: must be >= 5␣
↪and <=30.
query_list_size=15

# Name of the search help content.  Language resolution and file extension is handled␣
↪on service.
search_help=SearchHelp
```

## 16.27 logbook.ui

File ../../app/logbook/ui/src/main/resources/log_ui_preferences.properties:

```
# ------------------------------
# Package org.phoebus.logbook.ui
# ------------------------------

# Comma-separated list of default logbooks for new log entries.
default_logbooks=Scratch Pad

# The default query for logbook applications
default_logbook_query=search=*&start=12 hours&end=now

# Whether or not to save user credentials to file so they only have to be entered␣
↪once when making log entries.
save_credentials=false

# Stylesheet for the items in the log calendar view
calendar_view_item_stylesheet=Agenda.css

# Text to render for the "Level" field of a log entry. Sites may wish to customize␣
↪this with respect to
# its wording and its implied purpose.
level_field_name=Level:
```

## 16.28 olog.api

File ../../app/logbook/olog/client/src/main/resources/olog_preferences.properties:

```
# --------------------------------------
# Package org.phoebus.olog.api
# --------------------------------------

# The olog url
```

(continues on next page)

(continued from previous page)

```
olog_url=localhost:9092

# User credentials for olog
username=user
password=****

# Enable debugging of http request and resposnsed
debug=false

# The connection timeout for the Jersey client, in ms. 0 = infinite.
connectTimeout=0
```

## 16.29 olog.es.api

File ../../app/logbook/olog/client-es/src/main/resources/olog_es_preferences.properties:

```
# --------------------------------------
# Package org.phoebus.olog.es.api
# --------------------------------------

# The olog url
olog_url=http://localhost:8080/Olog

# User credentials for olog
username=admin
password=1234

# Enable debugging of http request and responses
debug=false

# The connection timeout for the Jersey client, in ms. 0 = infinite.
connectTimeout=0

# Comma separated list of "Levels" in the create logbook entry UI.
# Sites may wish to customize (and localize) this.
levels=Urgent,Suggestion,Info,Request,Problem
```

## 16.30 pv

File ../../core/pv/src/main/resources/pv_preferences.properties:

```
# ----------------------
# Package org.phoebus.pv
# ----------------------

# Default PV Type
default=ca
```

## 16.31 pv.ca

File ../../core/pv/src/main/resources/pv_ca_preferences.properties:

```
# -------------------------
# Package org.phoebus.pv.ca
# -------------------------

# Channel Access address list
addr_list=

auto_addr_list=true

max_array_bytes=100000000

server_port=5064

repeater_port=5065

beacon_period=15

connection_timeout=30

# Support variable length arrays?
# auto, true, false
variable_length_array=auto

# Connect at lower priority for arrays
# with more elements than this threshold
large_array_threshold= 100000

# Is the DBE_PROPERTY subscription supported
# to monitor for changes in units, limits etc?
dbe_property_supported=false

# Mask to use for subscriptions
# VALUE, ALARM, ARCHIVE
monitor_mask=VALUE

# Name server list
name_servers=
```

## 16.32 pv.formula

File ../../core/pv/src/main/resources/pv_formula_preferences.properties:

```
# ------------------------------
# Package org.phoebus.pv.formula
# ------------------------------

# Update throttle for input PVs
throttle_ms=500
```

## 16.33 pv.mqtt

File ../../core/pv/src/main/resources/pv_mqtt_preferences.properties:

```
# ---------------------------
# Package org.phoebus.pv.mqtt
# ---------------------------

# MQTT Broker
# All "mqtt://some/tag" PVs will use this broker
mqtt_broker=tcp://localhost:1883
```

## 16.34 pv.pva

File ../../core/pv/src/main/resources/pv_pva_preferences.properties:

```
# -------------------------
# Package org.phoebus.pv.pva
# -------------------------
# By default, these preference settings are empty,
# and the PVA library will then honor the commonly used
# environment variables like EPICS_PVA_ADDR_LIST,
# EPICS_PVA_AUTO_ADDR_LIST etc.
# Defining preference values will override the environment
# variables which allows consolidating PVA settings
# with all the CS-Studio preference settings.
#
#
# Network clients typically need to configure the first
# three settings to successfully connect to PVA servers
# on the local network.

# PVAccess address list
epics_pva_addr_list

# PVAccess auto address list - true/false
epics_pva_auto_addr_list

# Name servers used for TCP name resolution
epics_pva_name_servers

# The following parameters should best be left
# at their default.
#
# For details, see PVASettings in PV Access library.

# Port used for UDP name searches and beacons
epics_pva_broadcast_port

# PV server's first TCP port
epics_pva_server_port

# Connection timeout in seconds
epics_pva_conn_tmo
```

(continues on next page)

(continued from previous page)

```
# Maximum number of array elements shown when printing data
epics_pva_max_array_formatting

# TCP buffer size for sending data
epics_pva_send_buffer_size

# Timeout used by plain "put" type of write
# when checking success or failure.
# Note this is not used with asyncWrite,
# the "put-callback" which returns a Future
# for awaiting the completion,
# but only with the plain "put" that returns ASAP
epics_pva_write_reply_timeout_ms=1000
```

## 16.35 pvtable

File ../../app/pvtable/src/main/resources/pv_table_preferences.properties:

```
# ----------------------------------------
# Package org.phoebus.applications.pvtable
# ----------------------------------------

# Should all BYTE[] values be considered "long strings"
treat_byte_array_as_string=true

# Show the units when displaying values?
show_units=true

# Show a "Description" column that reads xxx.DESC?
show_description=true

# Default tolerance for newly added items
tolerance=0.1

# Maximum update period for PVs in millisecs
max_update_period=500
```

## 16.36 pvtree

File ../../app/pvtree/src/main/resources/pv_tree_preferences.properties:

```
# ---------------------------------------
# Package org.phoebus.applications.pvtree
# ---------------------------------------

# The channel access DBR_STRING has a length limit of 40 chars.
# Since EPICS base R3.14.11, reading fields with an added '$' returns
# their value as a char[] without length limitation.
# For older IOCs, this will however fail, so set this option
# only if all IOCs are at least version R3.14.11
read_long_fields=true
```

(continues on next page)

(continued from previous page)

```
# For each record type, list the fields to read and trace as 'links'.
#  Format: record_type (field1, field2) ; record_type (...)
#
# Fields can simply be listed as 'INP', 'DOL'.
# The syntax INPA-L is a shortcut for INPA, INPB, INPC, ..., INPL
# The syntax INP001-128 is a shortcut for INP001, INP002, ..., INP128
# The general syntax is "FIELDxxx-yyy",
# where "xxx" and "yyy" are the initial and final value.
# "xxx" and "yyy" need to be of the same length, i.e. "1-9" or "01-42", NOT "1-42".
# For characters, only single-char "A-Z" is supported, NOT "AA-ZZ",
# where it's also unclear if that should turn into AA, AB, AC, .., AZ, BA, BB, BC, ..,
↪ ZZ
# or AA, BB, .., ZZ
#
# bigASub is a CSIRO/ASCAP record type, doesn't hurt to add that to the shared␣
↪configuration
#
# scalcout is a bit unfortunate since there is no shortcut for INAA-INLL.
#
# alarm record has INP1-10. 1-9 handled by pattern, INP10 listed

fields=aai(INP);ai(INP);bi(INP);compress(INP);longin(INP);int64in(INP);mbbi(INP);
↪mbbiDirect(INP);mbboDirect(INP);stringin(INP);lsi(INP);subArray(INP);waveform(INP);
↪aao(DOL);ao(DOL);bo(DOL);fanout(DOL);longout(DOL);int64out(DOL);mbbo(DOL);
↪stringout(DOL);sub(INPA-L);genSub(INPA-L);calc(INPA-L);calcout(INPA-L);aSub(INPA-U);
↪seq(SELN);bigASub(INP001-128);scalcout(INPA-L,INAA,INBB,INCC,INDD,INEE,INFF,INGG,
↪INHH,INII,INJJ,INKK,INLL);alarm(INP1-9,INP10)


# Max update period in seconds
update_period=0.5
```

## 16.37 saveandrestore

File ../../app/save-and-restore/app/src/main/resources/save_and_restore_preferences.properties:

```
# ------------------------------------------------
# Package org.phoebus.applications.saveandrestore
# ------------------------------------------------

# Sort snapshots in reverse order of created time. Last item comes first.
sortSnapshotsTimeReversed=false

# Specify hierarchy parser class to enable TreeTableView in snapshot
# Hierarchy parser class should be in ui/snapshot/hierarchyparser
# RegexHierarchyParser is provided for convenience. Use , as separator for each regex␣
↪pattern.
# First matched pattern is used to create its hierarchy.
tree_tableview_enable=true
treeTableView.hierarchyParser=RegexHierarchyParser
regexHierarchyParser.regexList=(\\w+)_(\\w+):(\\w+)_(\\w+):(.*),(\\w+)_(\\w+):(\\w+)_
↪(.*),(\\w+)_(\\w+):(.*),(\\w+):(.*)
```

(continues on next page)

(continued from previous page)

```
# Read timeout (in ms) when taking snapshot
readTimeout=5000

# Limit used in "paginated" search, i.e. the number of search results per page
search_result_page_size=30

# Default save-and-restore search query. Used unless a saved query is located.
default_search_query=tags=golden

# If declared add a date automatically in the name of the snapshot "Take Snapshot"
#default_snapshot_name_date_format=yyyy-MM-dd HH:mm:ss
```

## 16.38 scan.client

File ../../app/scan/client/src/main/resources/scan_client_preferences.properties:

```
# ----------------------------------------
# Package org.csstudio.scan.client
# ----------------------------------------

# Name of host where scan server is running
host=localhost

# TCP port of scan server REST interface
port=4810

# Poll period [millisecs] of the scan client (scan monitor, plot, ...)
poll_period=1000
```

## 16.39 scan.ui

File ../../app/scan/ui/src/main/resources/scan_ui_preferences.properties:

```
# ----------------------------
# Package org.csstudio.scan.ui
# ----------------------------

# Show scan monitor status bar?
monitor_status=false
```

## 16.40 security

File ../../core/security/src/main/resources/phoebus_security_preferences.properties:

```
# ----------------------------
# Package org.phoebus.security
# ----------------------------

# Authorization file
```

(continues on next page)

(continued from previous page)

```
#
# If left empty, the built-in core/security/authorization.conf is used.
#
# When specifying a plain file name like "authorization.conf",
# the install location (Locations.install()) is searched for that file name.
#
# The file name can also be an absolute path like /some/path/auth.conf.
#
# Finally, the file name may use a system property like $(auth_file)
# which in turn could be set to either BUILTIN, a file in the install location,
# or an absolute path.
#
# When set to an invalid file, the user will have no authorizations at all.

# Use built-in core/security/authorization.conf
authorization_file=

# Use authorization.conf in the install location
#authorization_file=authorization.conf

# Secure store underlying implementation.
# Can be 'FILE' or 'IN_MEMORY'
secure_store_target=FILE
```

## 16.41 trends.databrowser3

File ../../app/databrowser/src/main/resources/databrowser_preferences.properties:

```
# ----------------------------------------
# Package org.csstudio.trends.databrowser3
# ----------------------------------------

# Default auto scale value
# Possible values are: true to enable the automatic calculation of the min/max Y-axis,
↪ or false to use min/max fixed values.
use_auto_scale=false

# Default time span displayed in plot in seconds
time_span=3600

# Default scan period in seconds. 0 for 'monitor'
scan_period=0.0

# Default plot update period in seconds
update_period=3.0

# .. elements in live sample buffer
live_buffer_size=5000

# Default line width
line_width=2

# Opacity of 'area'
#   0%: Area totally transparent (invisible)
```

(continues on next page)

(continued from previous page)

```
#  20%: Area quite transparent
# 100%: Area uses  solid color
opacity=40

# Default trace type for newly created traces.
# Allowed values are defined by org.csstudio.trends.databrowser3.model.TraceType:
# AREA, ERROR_BARS, SINGLE_LINE, AREA_DIRECT, SINGLE_LINE_DIRECT, SQUARES, ...
trace_type=AREA

# Delay in milliseconds that delays archive requests when
# the user moves the time axis to avoid a flurry of archive requests
# while interactively zooming and panning
archive_fetch_delay=500

# Number of concurrent archive fetch requests.
# When more requests are necessary, the background jobs
# will wait until the previously submitted jobs complete,
# to limit the number of concurrent requests.
#
# Ideally, the number can be high, but to limit the number
# of concurrent requests to for example an RDB,
# this value can be lowered.
#
# Note that this does not apply to 'exporting' data
# in spreadsheet form, where data for N channels is still
# collected by reading from N concurrent archive readers.
concurrent_requests=1000

# Number of binned samples to request for optimized archive access.
# Negative values scale the display width,
# i.e. -3 means: 3 times Display pixel width.
plot_bins=-3

# Suggested data servers
# Format:  <url>*<url>|<name>
# List of URLs, separated by '*'.
# Each URL may be followed by an "|alias"
#
# RDB URLs
# jdbc:mysql://localhost/archive
#
# Archive Appliance
# pbraw\://arcapp01.site.org:17668/retrieval
#
# Channel Archiver Network Data Server
# xnds://localhost/archive/cgi/ArchiveDataServer.cgi
#
# Channel Archiver index file (binary) or index.xml (list of indices)
# cadf:/path/to/index
# cadf:/path/to/index.xml
urls=jdbc:mysql://localhost/archive|RDB*xnds://localhost/archive/cgi/
↪ArchiveDataServer.cgi

# Default data sources for newly added channels
# Format: Same as 'urls'
archives=jdbc:mysql://localhost/archive|RDB*xnds://localhost/archive/cgi/
↪ArchiveDataServer.cgi
```

(continues on next page)

(continued from previous page)

```
# When opening existing data browser plot,
# use archive data sources specified in the configuration file (original default)
# or ignore saved data sources and instead use the preference settings?
use_default_archives=false

# If there is an error in retrieving archived data,
# should that archive data source be dropped from the channel?
# This is meant to avoid needless queries to archives that cannot be accessed.
# Note that archive data sources which clearly report a channel as "not found"
# will still be dropped. This option only configures if data sources which
# return an error (cannot connect, ...) should be queried again for the given channel.
drop_failed_archives=true

# With EPICS IOCs from release 7 on, the PVs
# "xxx", "ca://xxx" and "pva://xxx" all refer
# to the same record "xxx" on the IOC.
#
# When the plot requests "pva://xxx", the archive might still
# trace that channel as "ca://xxx" or "xxx".
# Alternatively, the archive might already track the channel
# as "pva://xxx" while data browser plots still use "ca://xxx"
# or just "xxx".
# This preference setting instructs the data browser
# to try all equivalent variants. If any types are listed,
# just "xxx" without any prefix will also be checked in addition
# to the listed types.
#
# The default of setting of "ca, pva" supports the seamless
# transition between the key protocols.
#
# When `equivalent_pv_prefixes` is empty,
# the PV name is used as is without looking for any equivalent names.
equivalent_pv_prefixes=ca, pva

# Re-scale behavior when archived data arrives: NONE, STAGGER
archive_rescale=STAGGER

# Shortcuts offered in the Time Axis configuration
# Format:
# Text for shortcut,start_spec|Another shortcut,start_spec
time_span_shortcuts=30 Minutes,-30 min|1 Hour,-1 hour|12 Hours,-12 hour|1 Day,-1␣
↪days|7 Days,-7 days

#It is a path to the directory where the PLT files for WebDataBrowser are placed.
plt_repository=/opt/codac/opi/databrowser/

# Automatically refresh history data when the liver buffer is full
# This will prevent the horizontal lines in the shown data when the buffer
# is too small to cover the selected time range
automatic_history_refresh=true

# Scroll step, i.e. size of the 'jump' left when scrolling, in seconds.
# (was called 'future_buffer')
scroll_step = 5

# Display the trace names on the Value Axis
```

(continues on next page)

(continued from previous page)

```
# the default value is "true". "false" to not show the trace names on the Axis
use_trace_names = true

# Prompt / warn when trying to request raw data?
prompt_for_raw_data_request = true

# Prompt / warn when making trace invisible?
prompt_for_visibility = true

# Shortcuts offered in the Time Axis configuration
# Format:
# Text for shortcut,start_spec|Another shortcut,start_spec
time_span_shortcuts=30 Minutes,-30 min|1 Hour,-1 hour|12 Hours,-12 hour|1 Day,-1␣
↪days|7 Days,-7 days

# Determines if the plot runtime config dialog is supported. Defaults to false as the␣
↪Data Browser
# offers the same functionality through its configuration tabs.
config_dialog_supported=false
```

## 16.42 ui

File ../../core/ui/src/main/resources/phoebus_ui_preferences.properties:

```
# ----------------------
# Package org.phoebus.ui
# ----------------------

# Show the splash screen?
# Can also be set via '-splash' resp. '-nosplash' command line options
splash=true

# 'Welcome' URL
#
# When left empty, the built-in welcome.html resource is used.
# Site-specific products can set this to their desired URL,
# which may include Java system properties to bundle content
# with the product, for example
#  file:$(phoebus.install)/welcome_to_hawkins_labs.html
welcome=

# Default applications
#
# When there are multiple applications that handle
# a resource, the setting determines the one used by default.
#
# Format is comma-separated list with sub-text of default application names.
# For example, "run, exe" would pick "display_runtime" over "display_editor",
# and "foo_executor" over "foo_creator".
# The patterns "edit, creat" would inversely open the editor-type apps.
#
# This makes the display_runtime and the 3d_viewer default apps,
# using display_editor and a potentially configured text editor for *.shp files␣
↪secondary
```

(continues on next page)

(continued from previous page)

```
default_apps=run,3d,convert_edm

# Hide SPI-provided menu entries
# Comma-separated list of class names
hide_spi_menu=org.phoebus.ui.monitoring.FreezeUI

# Top resources to show in "File" menu and toolbar
#
# Format:
# uri1 | uri2,Display name 2 | uri3,Display name 3
top_resources=examples:/01_main.bob?app=display_runtime,Example Display | pv://?sim://
↪sine&app=probe,Probe Example | pv://?sim://sine&loc://x(10)&app=pv_table,PV Table␣
↪Example | http://www.google.com?app=web, Google

# Home display file. "Home display" button will navigate to this display.
home_display=examples:/01_main.bob?app=display_runtime,Example Display

# How many array elements to show when formatting as text?
max_array_formatting=256

# UI Responsiveness Monitor Period
# Period between tests [millisec],
# i.e. the minimum detected UI freeze duration
# Set to 0 to disable
ui_monitor_period=500

# Show user ID in status bar?
status_show_user=true

# Set default save path
default_save_path=

# Set the path to a folder with default layouts
layout_dir=

# Compute print scaling in 'landscape' mode?
# Landscape mode is generally most suited for printouts
# of displays or plots, because the monitor tends to be 'wide'.
# At least on Mac OS X, however, the printing always appears to use
# portrait mode, so print layouts computed in landscape mode
# get cropped.
# Details can also depend on the printer driver.
print_landscape=true

# Color for text and the background for 'OK' alarm severity (R,G,B or R,G,B,A values␣
↪in range 0..255)
ok_severity_text_color=0,255,0
ok_severity_background_color=255,255,255

# Color for text and the background for 'MINOR' alarm severity
minor_severity_text_color=255,128,0
minor_severity_background_color=255,255,255

# Color for text and the background for 'MAJOR' alarm severity
major_severity_text_color=255,0,0
major_severity_background_color=255,255,255
```

(continues on next page)

(continued from previous page)

```
# Color for text and the background for 'INVALID' alarm severity
invalid_severity_text_color=255,0,255
invalid_severity_background_color=255,255,255

# Color for text and the background for 'UNDEFINED' alarm severity
undefined_severity_text_color=200,0,200,200
undefined_severity_background_color=255,255,255

# Color Configuration for the application "Alarm Area Panel" (R,G,B or R,G,B,A values␣
↪in range 0..255):
alarm_area_panel_ok_severity_text_color=255,255,255
alarm_area_panel_ok_severity_background_color=0,255,0

alarm_area_panel_minor_severity_text_color=255,255,255
alarm_area_panel_minor_severity_background_color=255,128,0

alarm_area_panel_major_severity_text_color=255,255,255
alarm_area_panel_major_severity_background_color=255,0,0

alarm_area_panel_invalid_severity_text_color=255,255,255
alarm_area_panel_invalid_severity_background_color=255,0,255

alarm_area_panel_undefined_severity_text_color=192,192,192
alarm_area_panel_undefined_severity_background_color=200,0,200,200
```

## 16.43 update

File ../../app/update/src/main/resources/update_preferences.properties:

```
# ----------------------------------------
# Package org.phoebus.applications.update
# ----------------------------------------

# Time to wait [seconds] for update check
# to allow more important tools to start
delay=10

# Version time/date
#
# If the distribution found at the `update_url`
# is later than this date, an update will be performed.
#
# The updated distribution must contain a new value for
# the org.phoebus.applications.update/current_version setting.
#
# By for example publishing updates with a 'current_version'
# that's one month ahead, you can suppress minor updates
# for a month.
#
# Format: YYYY-MM-DD HH:MM
#current_version=2018-06-18 13:10
current_version=
```

(continues on next page)

(continued from previous page)

```
# Location where updates can be found
#
# The file:, http: or https: URL is checked.
# If it exists, and its modification time is after `current_version`,
# the updated distribution is downloaded
# and the current Locations.install() is replaced.
#
# Location may include system properties
# and $(arch) will be replaced by "linux", "mac" or "win"
# to allow locations specific to each architecture.
update_url=
# update_url=https://controlssoftware.sns.ornl.gov/css_phoebus/nightly/product-sns-
↪$(arch).zip

#gitlab_api_url=https://HOST/api/v4
#gitlab_project_id=
gitlab_package_name=phoebus-$(arch)
#gitlab_token=

# List of regular expressions, comma-separated, which will be
# removed from the ZIP file entry.
# If result is empty string, the entry is skipped.
#
# The update ZIP file can have various formats.
#
# Basic ZIP file:
#    phoebus-{site, version}/*
#
# => Remove 'phoebus-.*' from entry name
#    to install _content_ of zip into install_location
#    without creating yet another subdir
#
# ZIP that's packaged for Windows, including JDK:
#    product-sns-0.0.1/*
#    jdk/*
#
# => Remove 'product-sns-*' from entry name,
#    skip 'jdk'.
#
# ZIP that's packaged for Mac: Either
#    phoebus.app/product-sns-0.0.1/*  => Remove .../
#    phoebus.app/jdk/*                => Skip
#    phoebus.app/Contents/*           => Skip
# or:
#    CSS_Phoebus.app/product-sns-0.0.1/*  => Remove .../
#    CSS_Phoebus.app/jdk/*                => Skip
#    CSS_Phoebus.app/Contents/*           => Skip
#
# Example:
# phoebus\.app/  - Strip Mac "phoebus.app/" from entries
#                  so they look more like the Windows example
#
# phoebus-[^/]+/ - Strip phoebus product name from ZIP entry
#
# jdk/.*         - Remove complete jdk entry to skip it
removals=CSS_Phoebus\\.app/Contents/.*,CSS_Phoebus\\.app/,phoebus\\.app/Contents/.*,
↪phoebus\\.app/,phoebus-[^/]+/,product-[^/]+/,jdk/.*
```

## 16.44 viewer3d

File ../../app/3d-viewer/src/main/resources/3d_viewer_preferences.properties:

```
# --------------------------------
# Package org.phoebus.app.viewer3d
# --------------------------------

# Time out for reading from a URI
read_timeout=10000

# Default directory for the file chooser.
default_dir=$(user.home)

# Cone is approximated with these many faces.
# 3: Triangular base, most minimalistic
# 8: Looks pretty good
# Higher: Approaches circular base,
# but adds CPU & memory usage
# and doesn't really look much better
cone_faces=8
```

# CHAPTER 17

# Change Log

A list of API changes, major features, and bug fixes included in each release.

The list is compiled by phoebus developers to document the most important changes associated with each release. For minor bug fixes and patches please refer to github issues and PRs.

## 17.1 Release 4.6.5 (current development version)

Date: TBD

- Avoid deadlock in script compilation and execution.
- Logbook modules restructured.
- Optimized display builder file save implementation.
- Display builder and scan editor undo stack maintained after file save.
- Logbook search cancellable.
- Fix for channel finder property editor.
- Scan infor parser performance improvement.
- PV race condition fix.
- Highlight overdrawn area of embedded display in edit mode.

## 17.2 Release 4.6.4

Date: Nov 16, 2020

**NOTE:** Bug in archiver settings handling causing the client to use default URL instead of the configured one.

- Alarm Datasource

- Simplification of the APIs for the *ContextMenuService*, *SelectionService*, and *AdapterService*.
- File browser context menu items to create new display or data browser plot.
- Array operations in formula functions.
- **Display Builder:**
  - The symbol widget shows a semi-transparent rectangle with the color "INVALID" for disconnected PVs.
  - The LED widget shows the label when the PV is disconnected and both the "On" and "Off" labels are the same.
- Statistics tab in Databrowser.
- PV status column (value, invalid, disconnected etc) in PV List.
- Number of undo/redo actions in display editor configurable (org.csstudio.display.builder.editor/undo_stack_size)
- **Save & Restore enhancements**
  - Set point may be edited prior to restore.
  - Display filter in snapshotData view.
  - Integration with Channel Finder.
  - Support for additional data types.
  - Extended server API to enable use of "file paths".

## 17.3 Release 4.6.3

Date: May 25, 2020

- Save & Restore application

## 17.4 Release 4.6.2

Date: Apr 27, 2020

## 17.5 Release 4.6.1

Date: Feb 4, 2020

## 17.6 Release 4.6.0

Date: Nov 28, 2019

- First release of phoebus framework

CHAPTER 18

# Docker

In an effort to ensure consistency of deployment, testing and build environments, we have a Dockerfile that developers may use to build a docker image. The following instructions assume that the developer is using linux and has Docker installed:

```
cd misc/
sudo docker build -t phoebus:latest .
sudo docker run -it phoebus:latest
```

From another shell(make sure to leave the previous shell open):

```
sudo docker cp phoebus [CONTAINER_ID]:/
```

You may also pull the following image from docker hub if you don't want to build the image yourself:

```
sudo docker pull lgomezwhl/phoebus-ci:latest
```

# CHAPTER 19

# Eclipse Debugging

Download Eclipse Oxygen 4.7.1a or later from http://download.eclipse.org/eclipse/downloads/

Start Eclipse like this:

```
export JAVA_HOME=/path/to/your/jdk-9-or-later
export PATH="$JAVA_HOME/bin:$PATH"
eclipse/eclipse -consoleLog
```

Check Eclipse Preferences:

```
Java, Installed JREs: JDK 9-or-later should be the default
Java, Compiler: JDK Compliance should be "9" or higher
```

Debugging with Eclipse

This assumes the project has been imported as a maven project into Eclipse(see instructions in README):

```
1. Open Eclipse
2. Go to `Run->External Tools->External Run COnfigurations`
3. Create a new `Program` configuration. Set location to `usr/bin/java` on linux.
   This is the location of the Java executable. For any other OS, it should not be
↪too hard
   to find that directory.
4. Set `Working Directory` to `phoebus/phoebus-product/target`.
5. Set arguments to:
```
--add-opens java.base/jdk.internal.misc=ALL-UNNAMED -Dio.netty.
↪tryReflectionSetAccessible=true
-Xdebug -Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=5005 -jar path_to_
↪repo/phoebus/phoebus-product/target/product-4.6.6-SNAPSHOT.jar
```
6. Click `Run`. The Eclipse console should output a port number. Write it down; we'll
↪use it for
   debugging later on.
7. Go to `Debug Configurations`
```

(continues on next page)

(continued from previous page)

```
8. Create a new `Remote Java Application`
9. Click on the `Source` tab and make sure all of the sub-modules/projects of the␣
↪phoebus project
   are checked. This will allow you to travel through source code when debugging code␣
↪in Eclipse.
10. For port, add the port from step 6.
11. Click `Debug`
```

Now this should connect to your JVM process you started on step 6 and you start debugging your code. Happy debugging!

CHAPTER 20

# GUI Testing

Automated UI testing is possible in phoebus with https://github.com/TestFX/TestFX

An example of UI testing may look like this:

```
@Test
public void TestNumberOfDockItems()
{
    Set<Node> menu = (Set<Node>) from(rootNode(Stage.getWindows().get(0))).
↪queryAllAs(Node.class);

    Node rootPane = menu.iterator().next();

    Assertions.assertThat(rootPane instanceof BorderPane).isTrue();

    BorderPane pane = (BorderPane) rootPane;

    Assertions.assertThat(pane.centerProperty().get() instanceof DockPane).isTrue();

    DockPane dockPane = (DockPane) pane.centerProperty().get();

    Assertions.assertThat(dockPane.getDockItems().size() == 2).isTrue();

    Assertions.assertThat(dockPane.getTabs().get(0) instanceof DockItem).isTrue();
    Assertions.assertThat(dockPane.getTabs().get(1) instanceof DockItem).isTrue();

    closePane();
    Assertions.assertThat((Stage.getWindows().size() == 0)).isTrue();
}
```

The snippet above is from "phoebus/core/ui/src/test/java/org/phoebus/ui/docking/SplitDockTestUI.java". TestFX has known issues such as incorrect behavior in headless mode and some unsupported nodes. For those issues you can follow their issue tracker https://github.com/TestFX/TestFX/issues

# CHAPTER 21

# Appendix

- genindex
- modindex
- search