

# Display Builder Tutorial

Jan. 2020

Kay Kasemir, [kasemirk@ornl.gov](mailto:kasemirk@ornl.gov)



# Changes

- Jan. 2019 USPAS: Initial version
- Jan. 2020 Class file details

# Display Builder

- Operator Interface Editor and Runtime
- Builds on ideas from EPICS edd/dm, medm, edm, ..
- Very compatible with CS-Studio 'BOY'
- Started ~2015 in CS-Studio/Eclipse, now in CS-Studio/Phoebus

**BL4A User Motors**

**Sample & Detector**

	Destination Pos	Current Pos	Status	Scan
SANGLE	0.5000 deg	0.5012 deg	●	Scan
SampleX	-8.9618	-8.9618	●	Scan
Beam Stop	0.0362 mm	0.0362 mm	●	Scan
Sample Changer	Undefined	-87.0008 mm	●	Scan
DANGLE	13.0000 deg	13.0015 deg	●	Scan

**Slits - Collimation**

	Destination Pos	Current Pos	Status	Scan
S1Width	0.500 mm	0.501 mm	●	Scan
S2Width	3.000 mm	2.996 mm	●	Scan
S3Width	0.500 mm	0.500 mm	●	Scan
S1Height	30.000 mm	29.998 mm	●	Scan
S2Height	30.000 mm	30.010 mm	●	Scan
S3Height	40.000 mm	40.000 mm	●	Scan

**Slits - Background**

	Destination Pos	Current Pos	Status	Scan
SSHWidth	0.000 mm	0.000 mm	●	Scan
L.Slit4	-1.0092 mm	-1.0145 mm	●	Scan
T.DetSlit	0.0539 mm	0.0521 mm	●	Scan
L.DetSlit	-0.1279 mm	-0.1276 mm	●	Scan
RSlit4	-58.5160 mm	-58.5165 mm	●	Scan
B.DetSlit	0.0438 mm	0.0455 mm	●	Scan
R.DetSlit	-4.9926 mm	-4.9927 mm	●	Scan

**Phoebus (on bl12-dassrv1.sns.gov)**

**Instrument Status**

- Beam Power (kW): 1.000000
- Primary Shutter: ●
- Secondary Shutter: ●
- Acquisition Software Status: ●
- Data Reduction Status: ●

**Proposal Information**

- Proposal #: IPTS-21677
- Proposal Title: Commissioning-TOPAZ
- Team Members: SXW,FCT,FTE,JIU (XCAMSUCAMS)

**Run Information**

- Scan Status: idle
- Run Status: idle
- Run Number: 31017
- Run Time: 2051.1 s
- Total Neutron Counts: 20017936
- Count Rate (counts/s): 0
- Total Proton Charge: 0.4585 C
- Beam Monitor 1 Counts: 2280665
- Beam Monitor 2 Counts: 1599739

**Main Detector XY Plot (4x4 Binned)**

Min: 400, Max: 185, Mean: 238.375, Total: 68652, Rate: 0 e/s

**Scan Monitor**

ID	Created	Name	State	%	Runtime	Finish	Command	Error
105	11:11:07	/tmp/20...	Aborted		02:04:30	13:15:38	-end-	Aborted
104	09:53:50	/tmp/20...	Aborted		01:16:45	11:10:35	-end-	Aborted
103	09:45:09	/tmp/20...	Aborted		00:02:02	09:47:12	-end-	Aborted
102	09:40:58	/tmp/20...	Aborted		00:02:26	09:43:25	-end-	Aborted
101	09:39:11	/tmp/20...	Aborted		00:01:08	09:40:20	-end-	Aborted
100	09:36:27	/tmp/20...	Aborted		00:02:18	09:38:45	-end-	Aborted
99	09:33:13	/tmp/20...	Aborted		00:02:57	09:36:11	-end-	Aborted
98	09:26:30	/tmp/20...	Aborted		00:04:16	09:30:47	-end-	Aborted
97	09:24:02	/tmp/20...	Aborted		00:01:35	09:25:38	-end-	Aborted

# Examples: SNS Accelerator

## SNS Central Control Room

Sep 13, 2018 10:56:05

Power on Target  
**1398 kW**

**Foil Image**

937 Turns

Energy: 1010 MeV

Rep Rate  
59.9 Hz

Beam  
Targ

**Target Image**

**Shutter Status**

1B NOMAD 2 BASIS 3 SNAP

4A Magnetism 4B Liquids 5 CNCS

12-Hr Power on Target

RTBT30 Beam Size

**Diagnostics**

Video Foil Monitor

Secondary

**Camera Setup**

Camera Select Primary Secondary

Overlay On Off

Primary Lamp Off On Off

Primary Insta-View Off On Off

Secondary Lamp Off On Off

Secondary Insta-View Off On Off

Calibrate Off On Off

**Neutral Density Filter**

Primary	Secondary
100% (blank)	100% (blank)
25%	25%
100% (blank)	100% (blank)
25%	25%

Horizontal		Vertical	
Amplitude	3756.96	Amplitude	3970.56
Mean	39.75 mm	Mean	37.62 mm
Std.Dev.	1.03 mm	Std.Dev.	0.75 mm
Offset	1.81 mm	Offset	1.85 mm
Slope	0.00	Slope	0.00
Area	0.00	Area	0.00

**Primary Camera Image:**

Plunge 13.70

Foil 800.00

## SNS E&RF Main Monitor

09/13/18 10:56:48

Power on Target  
**1377.20 kW**

Click LEDs for Detail

Rep Rate  
**59.9 Hz**

Avg Current  
**27.5 mA**

Recent Downtimes

2018/09/12 16:52:10	2018/09/12 16:54:25	0.0
2018/09/12 13:21:46	2018/09/12 13:22:37	0.0
2018/09/12 13:10:16	2018/09/12 13:14:01	0.1

Latest MPS Fault

Sep 13 2018 10:48:32 RFQ\_LLRF:HPM1:FPAR\_MEBT\_BS

RFQ	MEBT				DTL				CCL									
CAV RFQ	1	2	3	4	1	2	3	4	5	6	1	2	3	4				
MOD RFQ	1	2	3	4	RFQ	M3	M5	M1	M2	M3	M4							
XTMR RFQ					MEBT Amplifiers				X1	X2	X3	X4	X5	X6	X1	X2	X3	X4

XMTR	Modulator	Cavities																						
01	SCL 01	01a	01b	01c	02a	02b	02c	03a	03b	03c	04a	04b	05	SCL 05	04c	05a	05b	05c	06a	06b	06c	07a	07b	07c
09	SCL 09	08a	08b	08c	09a	09b	09c	10a	10b	10c	11a	11	SCL 12	11b	11c	12a	12b	12c	12d	13a	13b	13c	13d	
12	SCL 14	14a	14b	14c	14d	15a	15b	15c	15d	16a	16b	14	SCL 15	16c	16d	17a	17b	17c	17d	18a	18b	18c	18d	
17	SCL 18	19a	19b	19c	19d	20a	20b	20c	20d	21a	21b	17	SCL 21	21c	21d	22a	22b	22c	22d	23a	23b	23c	23d	

E&RF Systems (Mobile View)

HPRF Main HVCM Main LLRF Main

CS-Studio 'BOY' \*.opi files



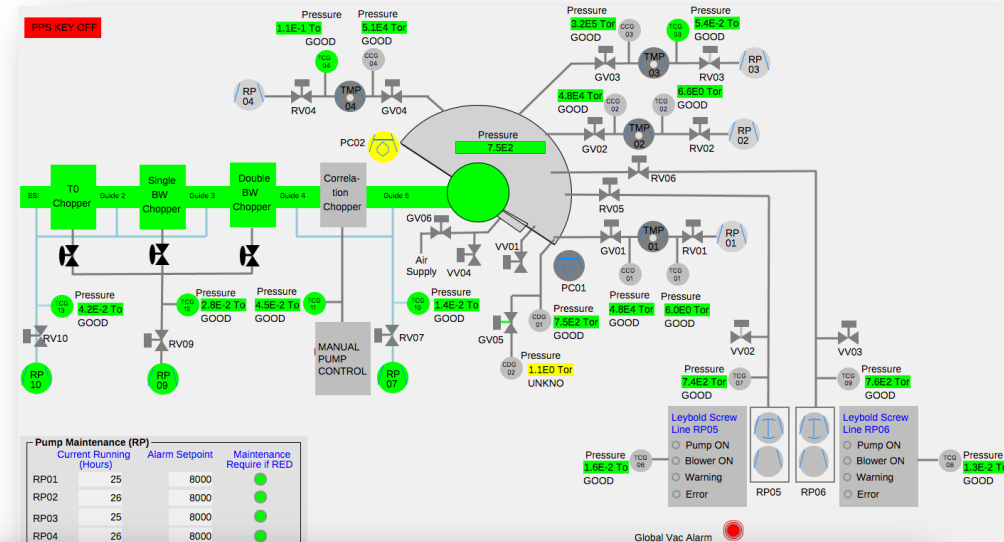
# Examples: SNS Beam Lines

Accelerator Mode: Target Power: 1398.96 kW Charge: 2.309E-5 C Energy: 1010.228 Mev Rate: 59.9 Hz

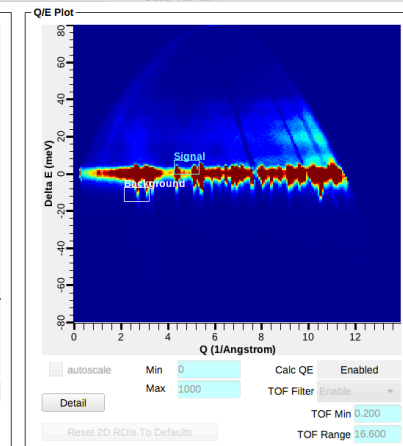
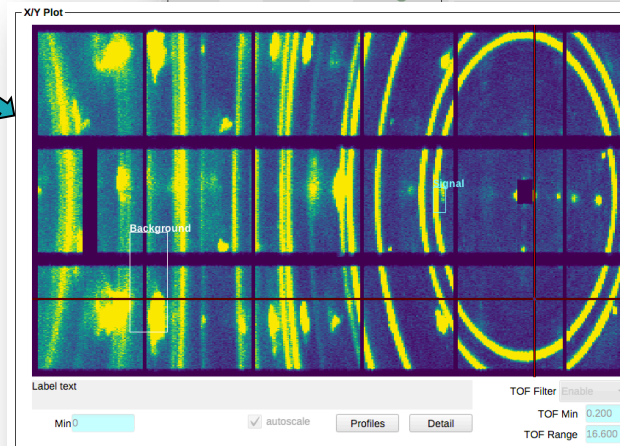
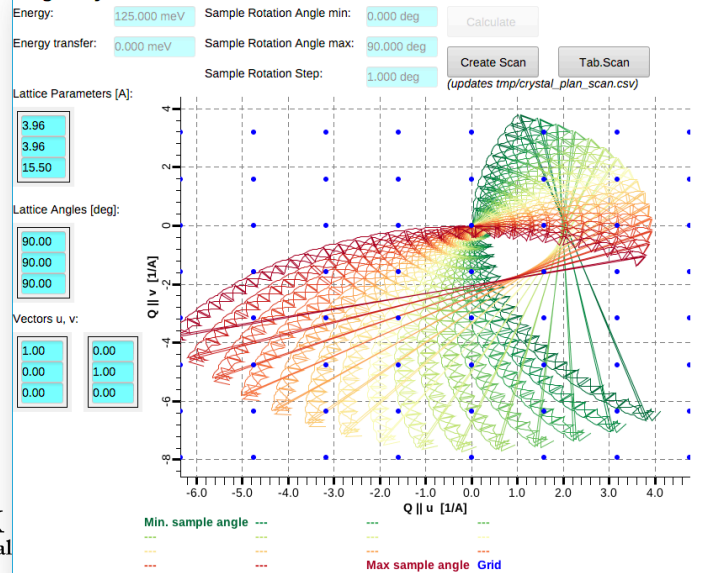
**HFIR**

<b>BL-1A USANS</b> Shutter: Run Scan Running Main TO Chopper IPPS	<b>BL-1B NOMAD</b> Shutter: Run Scan Finished Main TO Chopper Choppers IPPS	<b>BL-2 BASIS</b> Shutter: Run Scan Finished Main TO Chopper IPPS	<b>BL-3 SNAP</b> Shutter: Run Scan Finished Main TO Chopper IPPS	<b>BL-4A MRef</b> Shutter: Run Scan Aborted Main Choppers IPPS	<b>BL-4B LRef</b> Shutter: Run Scan Finished Main Choppers IPPS
<b>BL-5 CNCS</b> Shutter: Run Scan Running Main Choppers IPPS	<b>BL-6 EQ-SANS</b> Shutter: Run Scan Running Main Vacuum Choppers IPPS	<b>BL-7 VULCAN</b> Shutter: Run Scan Finished Main Detector Choppers IPPS	<b>BL-9 CORELLI</b> Shutter: Run Scan Paused Main Vacuum TO Chopper Choppers IPPS	<b>BL-10 VENUS</b> Shutter: Run Scan Running Main Choppers IPPS	<b>BL-11A POWGEN</b> Shutter: Run Scan Running Main TO Chopper Choppers IPPS
<b>BL-12 TOPAZ</b> Shutter: Run Main IPPS	<b>BL-13 FNBP</b> Shutter: Run Main Choppers IPPS	<b>BL-14B HYSPEC</b> Shutter: Run Scan Running Main Choppers IPPS	<b>BL-15 NSE</b> Shutter: Run Scan Running Main Choppers IPPS	<b>BL-16B VISION</b> Shutter: Run Scan Running Main Vacuum TO Chopper Choppers IPPS	<b>BL-17 SEQUOIA</b> Shutter: Run Scan Aborted Main Vacuum Choppers Detector/nED IPPS
<b>BL-18 ARCS</b> Shutter: Run Scan Running Main Vacuum Choppers IPPS					

Summaries: SE Case, CMF, NCL, Gateways, ODH, Instruments Data



## Single Crystal Planner



**X-Y ROI**

	Min	Max	Mean	Total	Total +/-	Rate
Signal	11	363	116.194	22774	150.911	0 e/s
Background	0	1425	71.161	160824	401.029	0 e/s
S/B	0.000	0.255	1.633	0.142	0.001	0.000

**Q-E ROI**

	Min	Max	Mean	Total	Total +/-	Rate
Signal	44	8996	778.156	105051	324.116	0 e/s
Background	6	2090	182.778	24675	157.083	0 e/s
S/B	7.333	4.304	4.257	4.257	0.030	0.000

**Data Collection**

Total Counts	4890418	0 e/s
Proton Charge	4.0063559E-1 C	
Beam Power	1402537 Watts	
Data Collection State	Idle	
Data Collection Pause	Not Paused	

**Q-E Axes & ROI Position Details**

	ROI Start	ROI Size	Start	End	Bin Size	
Q Signal	4.268	1.067	Q Axis	0.0000	13.9396	0.0697
E Signal	0.000	7.500	E Axis	-80.0000	80.0000	0.8000
Q Background	2.134	1.067				
E Background	-15.000	7.500				

# Browse the Examples

- Start CSS/Phoebus
- Your setup might have a menu entry
  - File, Top Resources, Examples
- If not, or if you'd like to inspect and edit the examples
  - Applications, Display, Examples, Install Example Displays

Tab  
Hover mouse,  
open Context Menu,  
Close

Example Display  
Push any of the buttons

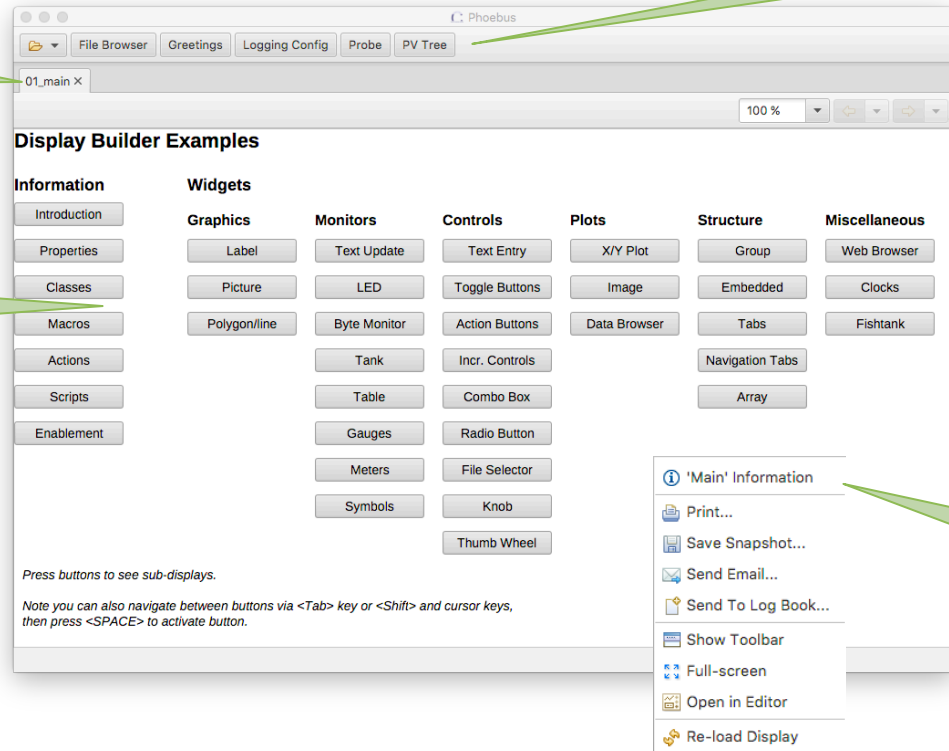
Main Application Toolbar  
Menu Window, Show Toolbar

Display Runtime Toolbar  
Context Menu Window,  
Show / Hide Toolbar

Navigate back/forward  
also via Alt-Left, Alt-Right cursor keys

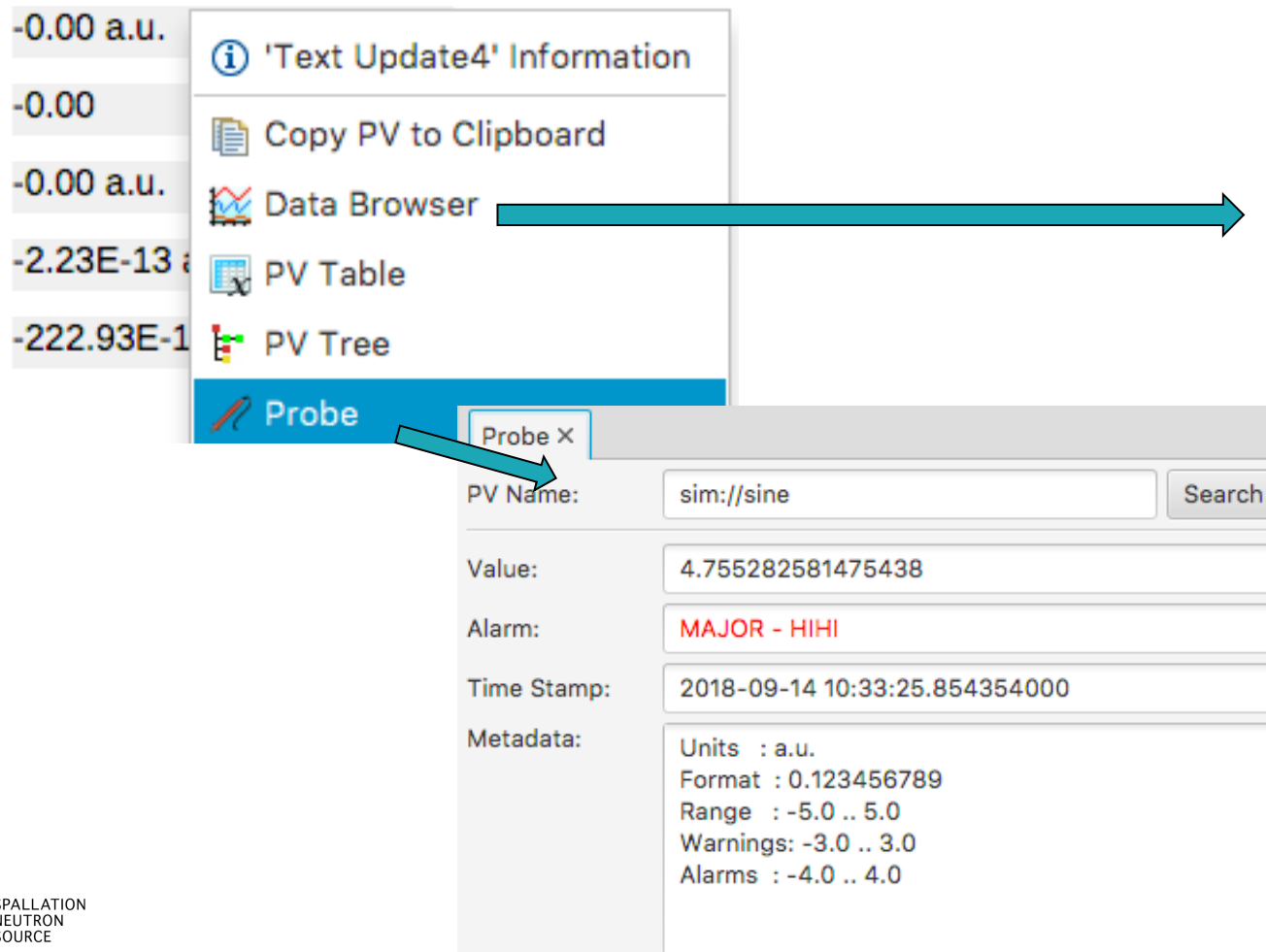
Zoom  
to view large control room displays on  
office computer

Context Menu  
Details change with widget  
on which menu was invoked



# Send PV to other Tools

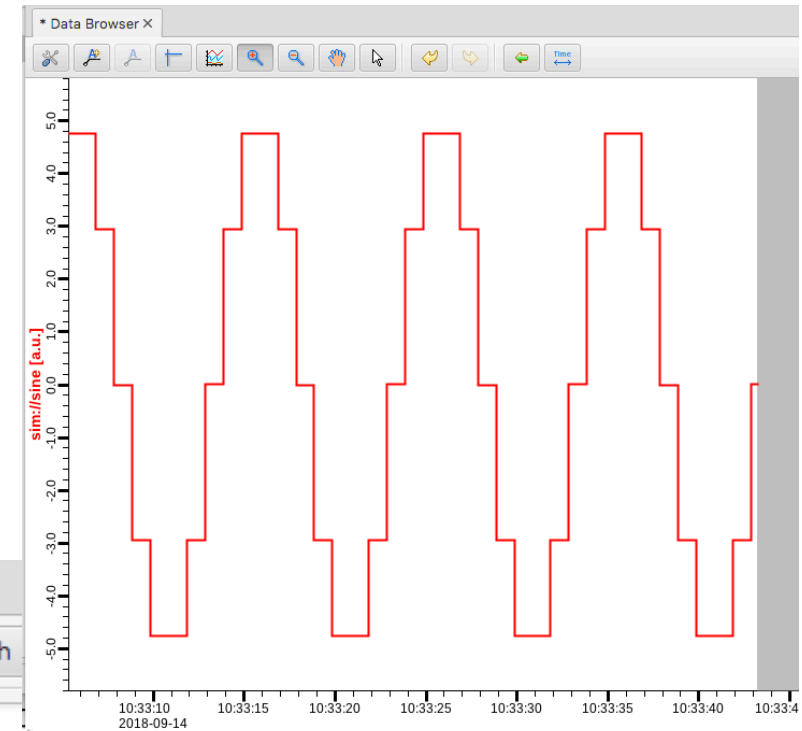
Context menu opens other tool with PV



The screenshot shows a context menu for a PV value. The menu items are:

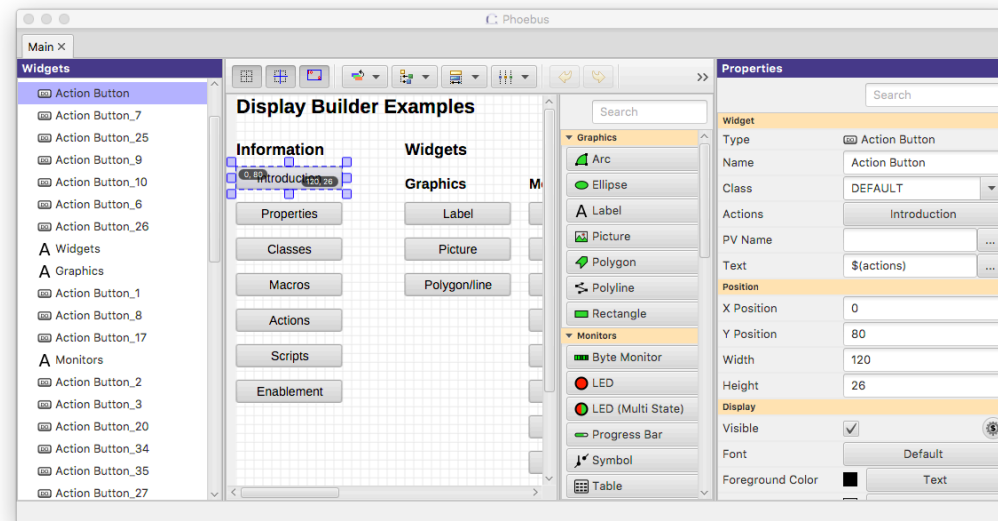
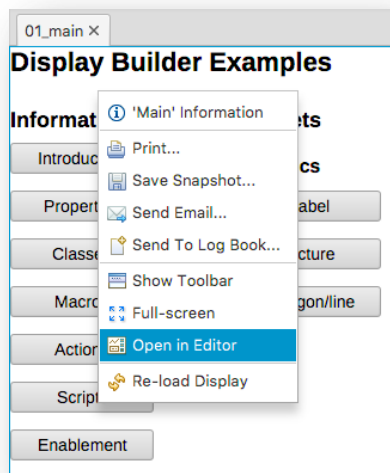
- 'Text Update4' Information
- Copy PV to Clipboard
- Data Browser
- PV Table
- PV Tree
- Probe

Arrows indicate that clicking 'Data Browser' leads to the 'Data Browser X' window, and clicking 'Probe' leads to the 'Probe X' window.

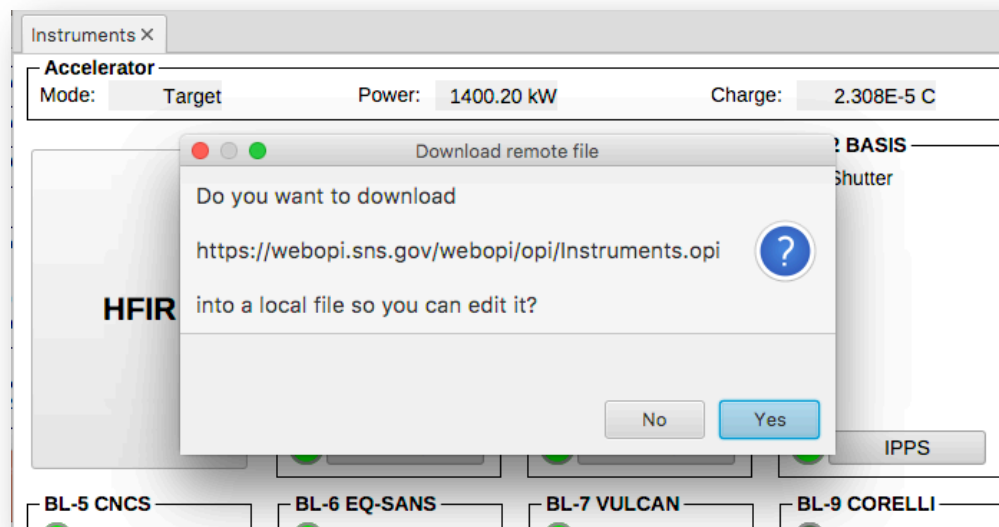


# Open Existing Display In Editor

- Context menu can open any display in Editor



- Downloads remote files



# Create New Display

Menu Applications, Display, New Display

- Enter a name with .bob file extension

Save & Execute the Display

Property Panel  
Edit properties of  
selected widgets

Main Editor Area

Select Widgets  
Move, resize widgets  
Ctrl-C, V, X to copy, paste, delete (⌘ on Mac)

The screenshot shows the LabVIEW 'Create New Display' dialog. On the left, a 'Widgets' list contains 'A Label', 'A Label\_1', '0.0 Text Update', and 'A Label\_2'. The '0.0 Text Update' widget is selected. The main editor area, titled 'My Display', shows a grid with the text 'Some Value: sim://sine Some comment.' and a 'Text Update' widget. The 'Properties' panel on the right shows settings for the selected widget: Type (0.0 TextUpdate), Name (Text Update), Class (DEFAULT), PV Name (sim://sine), Position (X: 100, Y: 50, Width: 100, Height: 20), and Display options (Visible checked, Font Default, Foreground Color Black, Background Color ReadBackground, Transparent unchecked, Format Default).



# Editing a Display

## Selecting Widgets

- a) Click single widget
- b) Ctrl-click to add widget (⌘ on Mac)
- c) Drag 'rubberband' around widgets
- d) Click or Ctrl/ ⌘ click in widget list

## Widget Palette

Drag widget into editor

- or -

- 1) Select Widget Type
- 2) Draw rectangular area in display

## Quick Edit

Double-click widget to

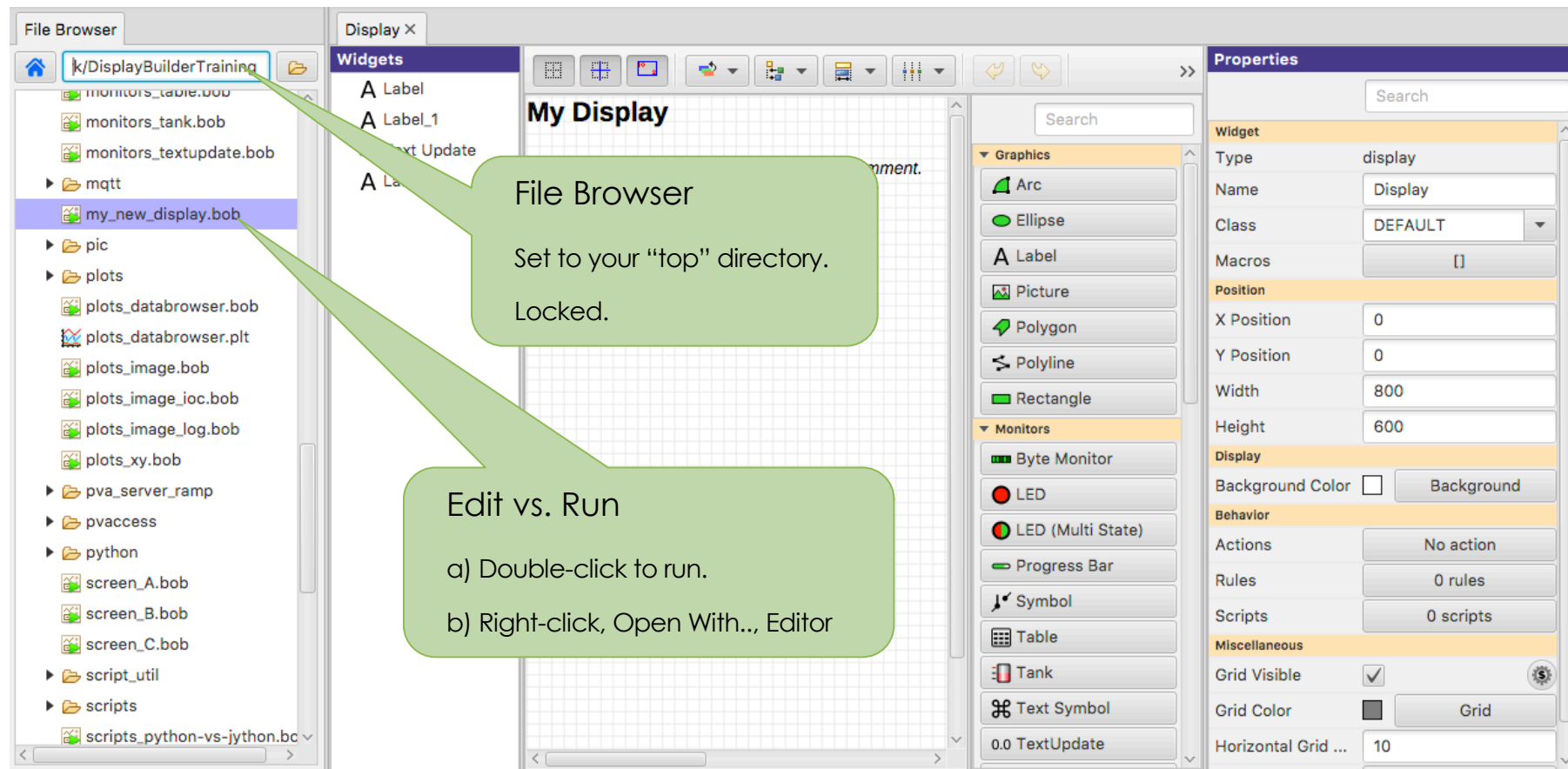
- a) Edit text of Label
- b) Edit PV of widgets that use a PV

The screenshot shows the LabVIEW graphical programming environment. The central workspace, titled 'My Display', contains a text widget with the text 'Some Value: sim://sine Some comment.'. The widget is highlighted with a blue border and small handles. To the left is the 'Widgets' palette, and to the right is the 'Widget Palette' and 'Properties' window. The 'Properties' window shows the following settings for the selected widget:

Widget	
Type	0.0 TextUpdate
Name	Text Update
Class	DEFAULT
PV Name	sim://sine
Position	
X Position	100
Y Position	50
Width	100
Height	20
Display	
Visible	<input checked="" type="checkbox"/>
Font	Default
Foreground Color	Text
Background Color	ReadBackground
Transparent	<input type="checkbox"/>
Format	Default

# Suggested Setup for Editing

- Pick a top directory, for example where you installed the example files
- Open Applications, Utility, File Browser
  - Set it to your top directory
  - On file browser tab, open context menu, “Split Horizontally”, then “Lock Pane”
- Menu Window, Save Layout As..
  - “Editing”
- Menu Applications, Display, New Display
  - Create new file in your top directory



# Keep It Simple

1. Add a Widget
2. Enter Label's Text or Widget's PV Name
3. Done

Basic Number:	-4.76 a.u.
Disconnecting channel:	<sim://intermittent>
Basic Text:	AAAAA

## At Runtime, widget will

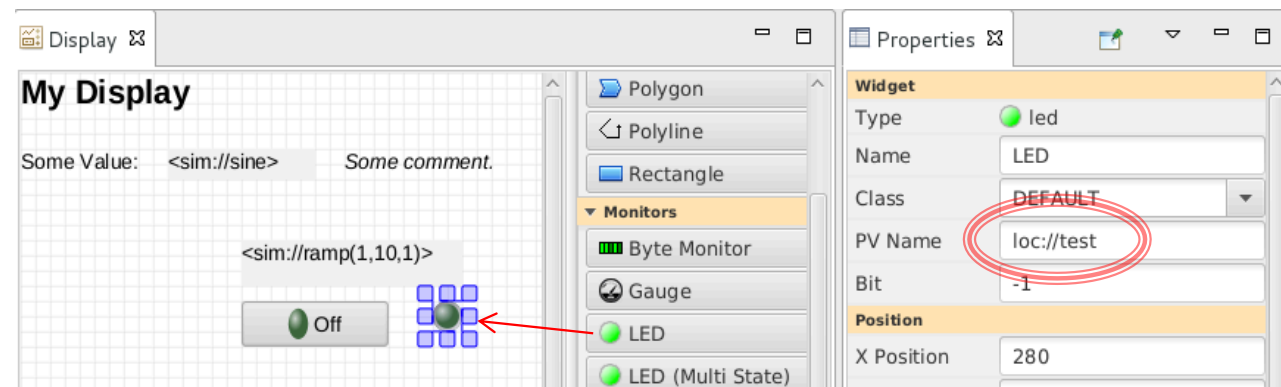
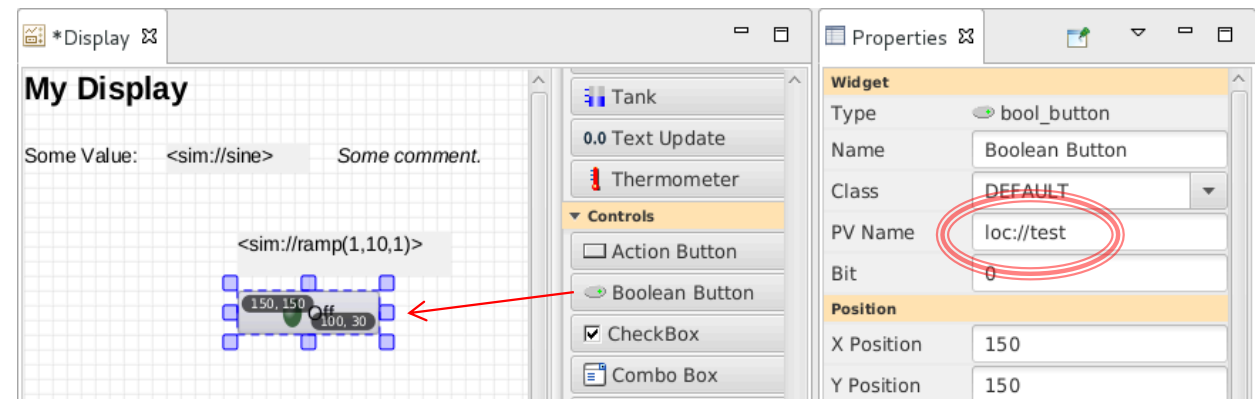
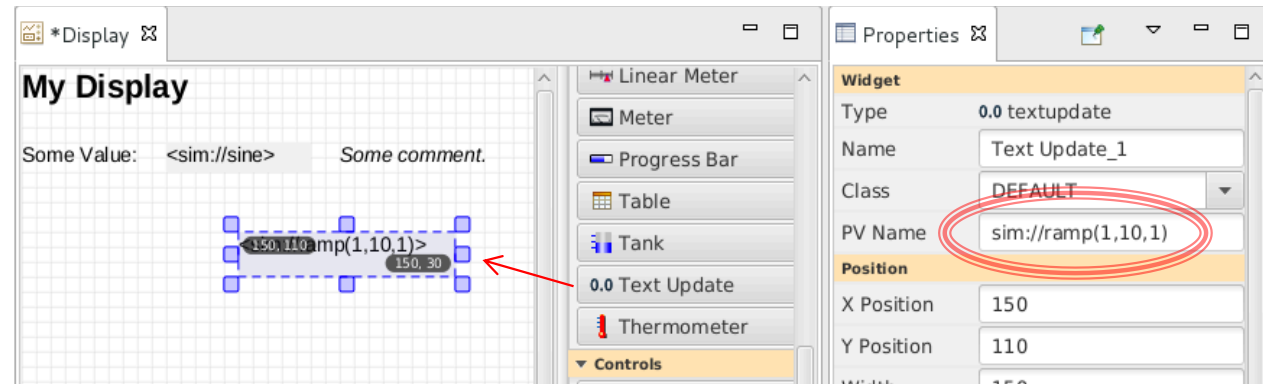
- ✓ Show PV's value, formatted, with units
- ✓ Indicate alarm, disconnect
- ✓ Show tool-tip with PV name and value
- ✓ Combo options read from Enum PV, slider range from numeric PV
- ✓ Disabled when 'control' widget has no PV write access

# Extend the First Display

- Drag a “Text Update” from the palette
  - Enter PV name “sim://ramp(1, 10, 1)”. Note PV name auto-completion popup.



- Add “Boolean Button”
  - PV name “loc://test”
- Add “LED”
  - PV name “loc://test”. Note name in PV History.
- Execute the display
  - Toolbar Button or Context Menu



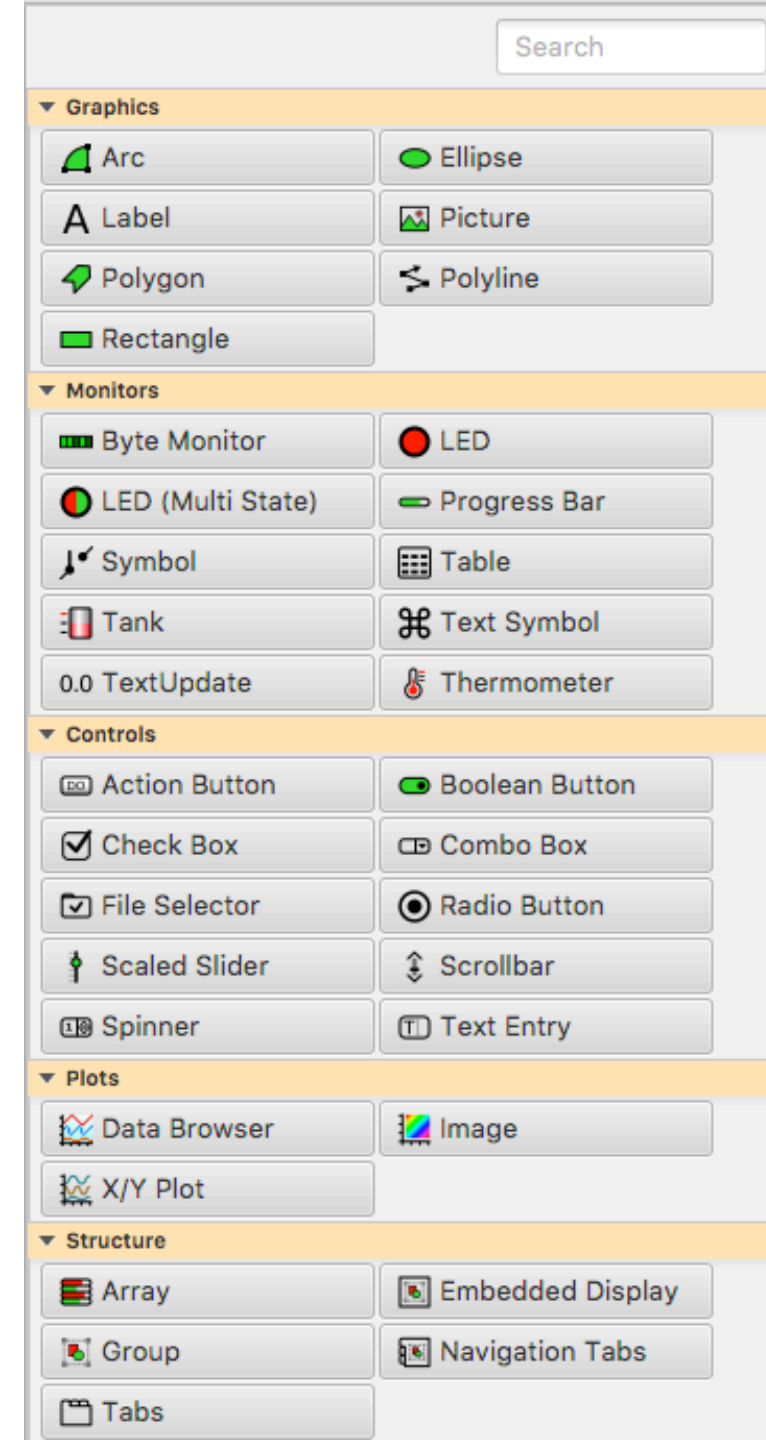
# PV Names

- `ca://some_pv_name`
  - EPICS Channel Access PV
- `some_pv_name`
  - Typically same, since “ca://” is the default
- `sim://sine`
  - Simulated PV. See auto-completion hints
- `loc://x(4)`
  - Local PV. See auto-completion hints
- `pva://x`
  - EPICS pvAccess



# Widget Palette

- Shows all available widgets
  - Enter name for "Search"
  - Hover mouse for description
  - Drag -or- Select & Rubberband
- Categories
  - Graphics show static label, picture, ..
  - Monitors update based on reading a PV
  - Controls read a PV and can write to the PV
  - Plots tend to read from one or more (waveform) PVs
  - Structures group widgets, embed sub-displays



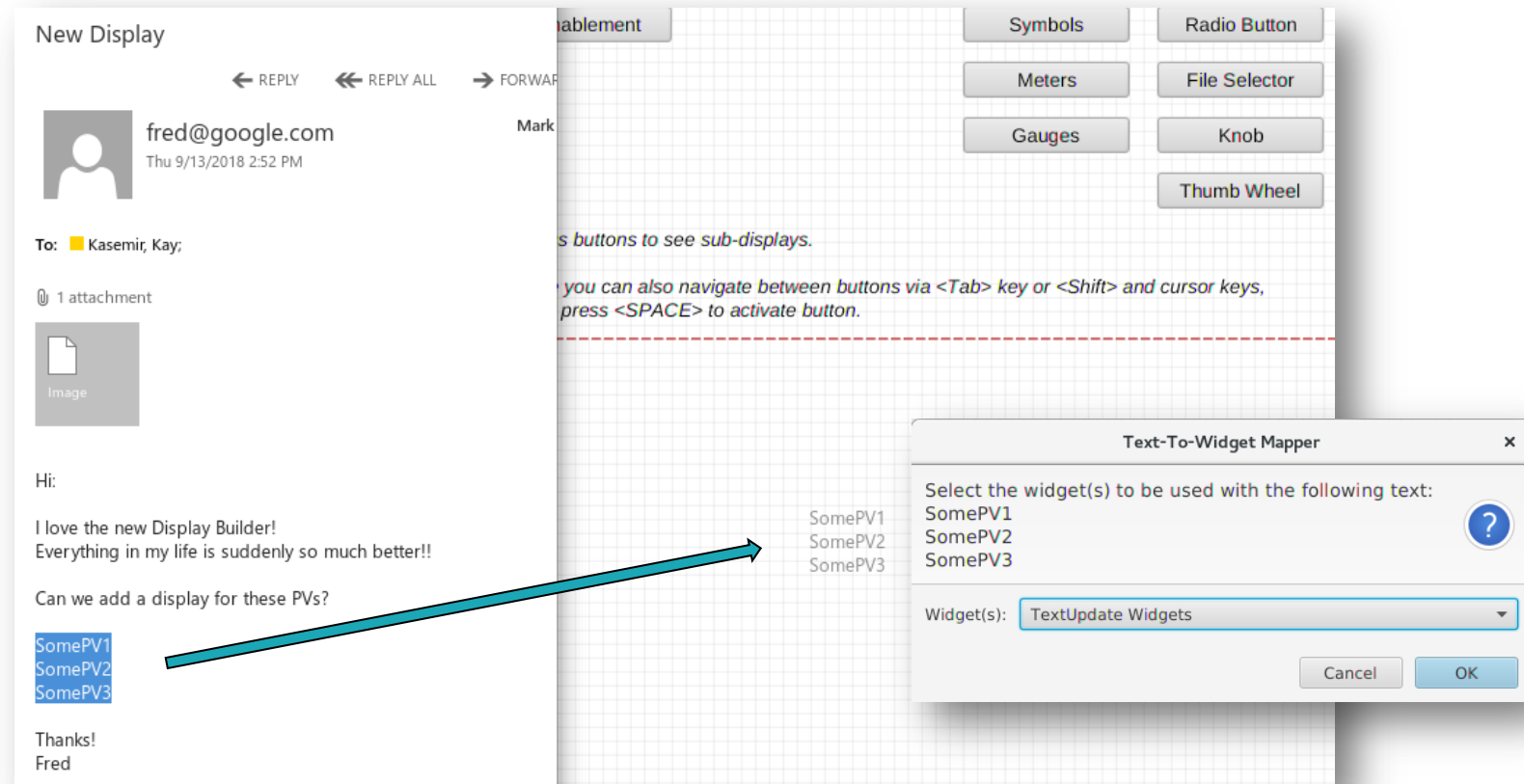
# Create Widgets via Drag/Drop from other Apps

## Email with list of PVs?

- Drag that text into Display Editor
- Select widget type

## Supported:

- Text → Label
- Text → PV Widget
- Image File → Picture Widget
- \*.bob File → Embedded Display Widget



# Manipulating Widgets

Snap to Grid

.. Other Widgets

Show Coordinates

Order Front/Back

Align

Size

Widget List

Select widgets

Rename Widgets

View/change their order.

Widgets

- A Label
- A Label\_1
- A Label\_2
- LED
- 0.0 Text Update
- ▼ Group
  - Boolean Button
  - 0.0 TextUpdate

My Display

Some Value: 100, 50 Some comment.

Group

Off

138, 148

Widget Manipulation Tools:

- Snap to Grid
- .. Other Widgets
- Show Coordinates
- Order Front/Back
- Align
- Size
- Distribute

Distribute

Selected Widgets Tracker

Move or resize selected widgets

# Display Properties

Click on display background to select no widget for editing overall display properties

- Name
  - Shown in Tab
- Macros
  - Used by all widgets in this display
- Grid size
  - Can aid with placing widgets

Properties	
Search	
<b>Widget</b>	
Type	display
Name	Display
Class	DEFAULT
Macros	[]
<b>Position</b>	
X Position	0
Y Position	0
Width	800
Height	600
<b>Display</b>	
Background Color	<input type="checkbox"/> Background
<b>Behavior</b>	
Actions	No action
Rules	0 rules
Scripts	0 scripts
<b>Miscellaneous</b>	
Grid Visible	<input checked="" type="checkbox"/>
Grid Color	<input type="checkbox"/> Grid
Horizontal Grid Step Size	10
Vertical Grid Step Size	10

# Widget Properties

Select one (or more) widgets to edit their (common) properties

- Search
  - To find desired property
- PV Name
  - Most important property for most widgets

Details depend on the widget type

The image shows a 'Properties' panel for a widget. The panel is organized into several sections:

- Widget**: Type (0.0 TextUpdate), Name (Text Update), Class (DEFAULT), PV Name (sim://sine).
- Position**: X Position (100), Y Position (50), Width (100), Height (20).
- Display**: Visible (checked), Font (Default), Foreground Color (black), Background Color (ReadBackground), Transparent (unchecked), Format (Default), Precision (-1), Show Units (checked), Horizontal Alignment (Left), Vertical Alignment (Top), Wrap Words (checked), Rotation (0 degrees).
- Behavior**: Actions (No action), Rules (0 rules), Scripts (0 scripts), Tool tip (\$(pv\_name)\$ (pv\_value)), Alarm Border (checked), Interactive (unchecked).
- Miscellaneous**: Border Width (0), Border Color (black).



# Common Widget Properties

Defaults tend to be reasonable:

- Format with precision set by PV
- Show units provided by PV
- Alarm-sensitive Border
- Fetch Items (Combo, ...) from PV

Instead of changing them,  
maybe the PV needs to be updated?

Still, can be adjusted as needed for the display.

The image shows a 'Properties' panel for a widget. The panel is organized into several sections:

- Widget:** Type: 0.0 TextUpdate; Name: Text Update; Class: DEFAULT; PV Name: sim://sine
- Position:** X Position: 100; Y Position: 50; Width: 100; Height: 20
- Display:** Visible: ; Font: Default; Foreground Color:  (black); Background Color:  ReadBackground; Transparent: ; Format: Default; Precision: -1; Show Units: ; Horizontal Alignment: Left; Vertical Alignment: Top; Wrap Words: ; Rotation: 0 degrees
- Behavior:** Actions: No action; Rules: 0 rules; Scripts: 0 scripts; Tool tip: \$(pv\_name)\$ (pv\_value); Alarm Border: ; Interactive:
- Miscellaneous:** Border Width: 0; Border Color:  (black)

# Predefined "Named" Colors and Fonts

Use whenever possible!

Name: Text Update\_4  
Class: DEFAULT  
PV Name: sim://sine

**Position**  
X Position: 191  
Y Position: 351  
Width: 170  
Height: 20

**Display**  
Visible:   
Font: Default  
Foreground Color: Text  
Background Color: ReadBackground  
Transparent:   
Format: Default  
Precision: -1  
Show Units:   
Horizontal Alignment: Left  
Vertical Alignment: Top  
Wrap Words:   
Rotation: 0 degrees

**Position**  
X Position: 191  
Y Position: 91  
Width: 170  
Height: 20

**Display**  
Visible:   
Font: Default  
Foreground Color: Text

Font – Select a predefined font and/or customize it.

Predefined Fonts: Comment, Default, Default Bold, Fine Print, Header 1, Header 2, Header 3, Oddball

Fonts Families: Liberation Sans, Liberation Serif, Libian SC, LiHei Pro, LiSong Pro, Lucida Bright, Lucida Grande, Lucida Sans, Lucida Sans Typewriter

Style: Regular    Size: 14.0

Preview: Example Text

Default    Cancel    OK

Foreground Color – Select a predefined color and/or customize it.

Predefined Colors: DISCONNECTED, Grid, Header\_Background, Header\_ForeGround, INVALID, MAJOR, MINOR, Off, OK, On, Read\_Background, STOP, Text, Write\_Background

Search: \_\_\_\_\_

Custom Color  
Color: Black  
Red: 0  
Green: 0  
Blue: 0  
Alpha: 255

original    default

Default    Cancel    OK

# Configuring Named Colors, Fonts

```
# -----  
# 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
```

Ideally set at start  
of project

```
color.def  
# Named colors  
#  
# Format:  
# NameOfColor = red, green, blue [, alpha ] |  
# PreviouslyDefinedNameOfColor  
# with values in 0..255 range.  
#  
# Whenever possible, use named colors in displays  
# instead of arbitrary red/green/blue values.  
  
# ----- Predefined colors -----  
# May be overridden in here  
  
# Alarm related  
OK = 0, 255, 0  
MINOR = 255, 128, 0  
MAJOR = 255, 0, 0  
INVALID = 255, 0, 255  
DISCONNECTED = 200, 0, 200, 200  
  
# Default color for text  
Text=0,0,0  
  
# Default color for 'active' text that's being edited  
ActiveText=255, 255, 0  
  
# Display background  
Background = 255, 255, 255  
  
# .. for widgets that read/write a value  
Read_Background = 240, 240, 240  
Write_Background = 128, 255, 255  
  
# .. for buttons  
Button_Background = 210, 210, 210  
  
# ----- Examples for additional colors -----  
  
# Also show ideas for site-specific guidelines that  
# are required to make sense of the color names.  
  
# Styling  
Header_Background=77,77,77  
Header_ForeGround=255,255,255  
  
# Use alarm colors only when you mean to indicate an
```

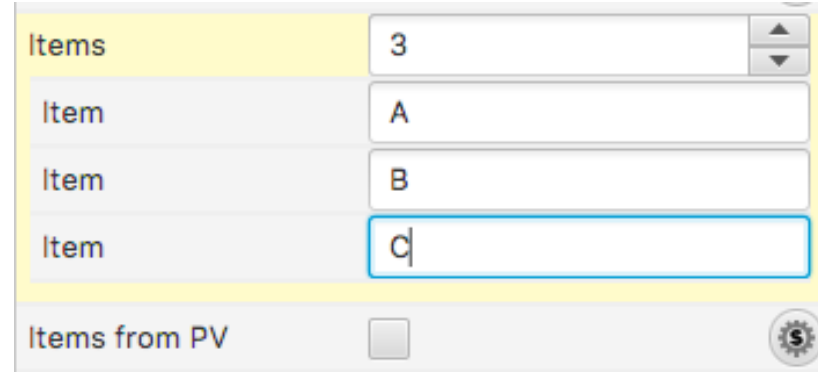
```
font.def  
// Named font definitions  
//  
// Entries in this file are read in sequence.  
// A later entry in the file can override  
// an earlier entry in the file.  
//  
// In a production setup, this file may be constructed  
// by concatenating a generic file with a more specific file,  
// and the specific entries would then override generic entries  
// of the same name.  
  
// Format:  
//  
// NamedFont['(' OS ')'] = Family '-' Style '-' Size | '@'PreviouslyDefinedNamedFont  
//  
// Family: Font family name "Liberation Sans", "Liberation Mono", "Liberation Serif"  
// Style: "regular", "bold", "italic", "bold italic"  
// Size: Font height in pixels  
// OS: "windows", "linux", "macosx"  
//  
// Leading/trailing spaces around each element are OK, but if the font family  
// is "Liberation Sans", it has to be typed with just that one space between  
// "Liberation" and "Sans"  
//  
// Examples of named fonts  
//  
// Default = Liberation Sans - regular - 14  
// Default Bold = Liberation Sans - bold - 14  
// Header 1 = @Default Bold  
//  
// Speaking of "Liberation Sans":  
// The display builder includes the "Liberation" fonts  
// from https://fedorahosted.org/liberation-fonts.  
// Their use is encouraged because the resulting displays  
// will always render correctly.  
// When using other fonts, for example "Arial" on Windows,  
// the font might not be available to a display builder  
// runtime that is executing on Mac OS or Linux.  
  
// Predefined fonts that this file could re-define  
Default = Liberation Sans - regular - 14  
Default Bold = Liberation Sans - bold - 14  
Header 1 = Liberation Sans - bold - 22  
Header 2 = Liberation Sans - bold - 18  
Header 3 = Liberation Sans - bold - 16  
Comment = Liberation Sans - italic - 14
```

# Widget Notes

- Text Entry, Text Update:
  - Set Format = String for “long string” waveforms. Default will show array.
- LED, Boolean Button, Checkbox
  - Boolean PV
  - Numeric PV 0 or not 0 (when “Bit” set to default of -1)
  - Bit in a numeric PV (when “Bit” set to 0, 1, 2, ...)
- Multi-State LED
  - Enumerated or numeric PVs
  - Defaults to using state values 0, 1, 2, 3, ..

# Widget Notes

- Combo Box, Radio Button:
  - Best for enumerated PV: Enter PV name, done
  - Alternatively, un-check “Items from PV” and enter items



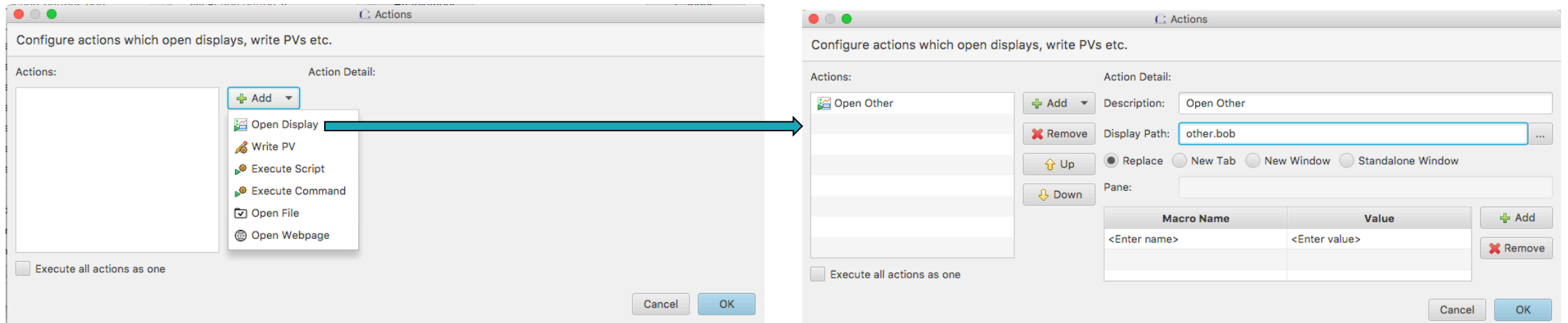
The screenshot shows a widget interface with the following elements:

- A yellow header bar with the label "Items" and a numeric input field containing the value "3".
- Three input fields labeled "Item" containing the values "A", "B", and "c". The "c" field is highlighted with a blue border.
- A checkbox labeled "Items from PV" which is currently unchecked.
- A gear icon in the bottom right corner of the widget.



# Action Button

1. Add ActionButton
2. Configure “Actions” property, add “Open Display”



3. Run: Clicking button opens the “other” display.

*In principle, any widget can have ‘Actions’.  
They appear in the widget’s runtime context menu.  
But it’s not obvious to end users that for example a Label will have actions.*

# Screen Navigation

- Replace

- Suggested default.
- Allows back/forward navigation as in web browser



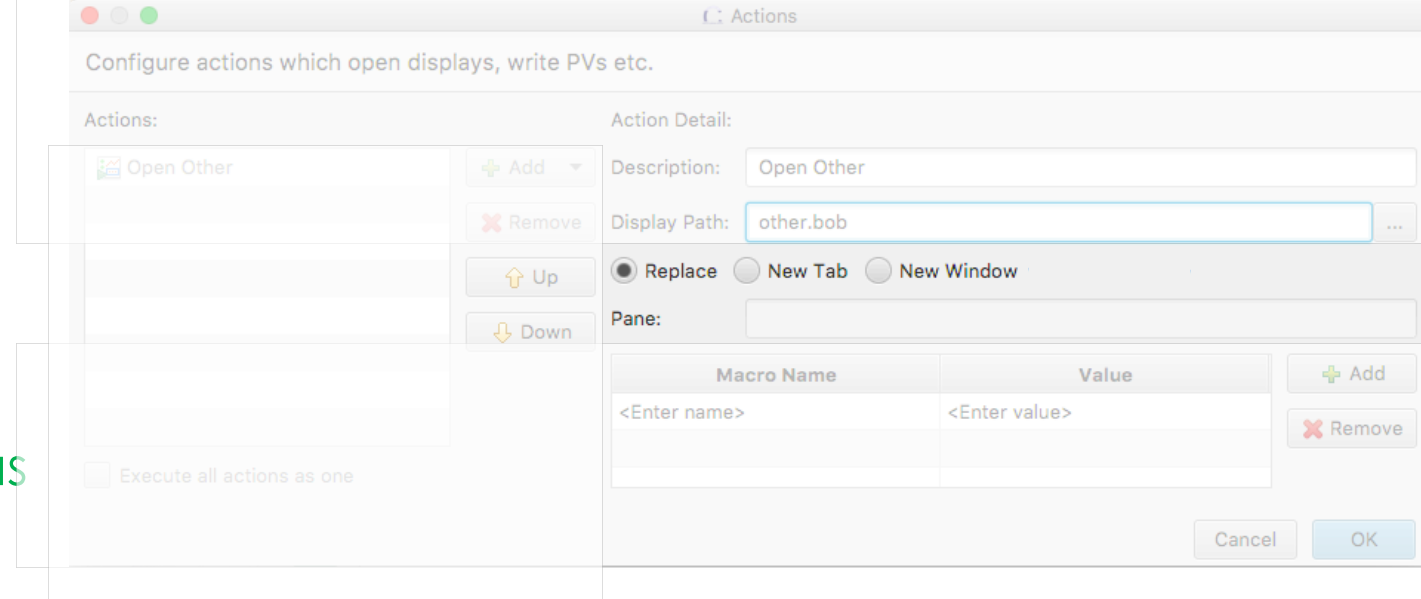
- Minimizes number of open screens

- New Tab

- Opens in new tab
- Allows specific Pane name

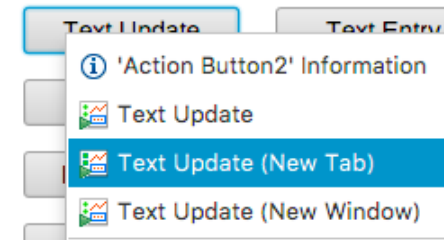
- New Window

- Opens in new window



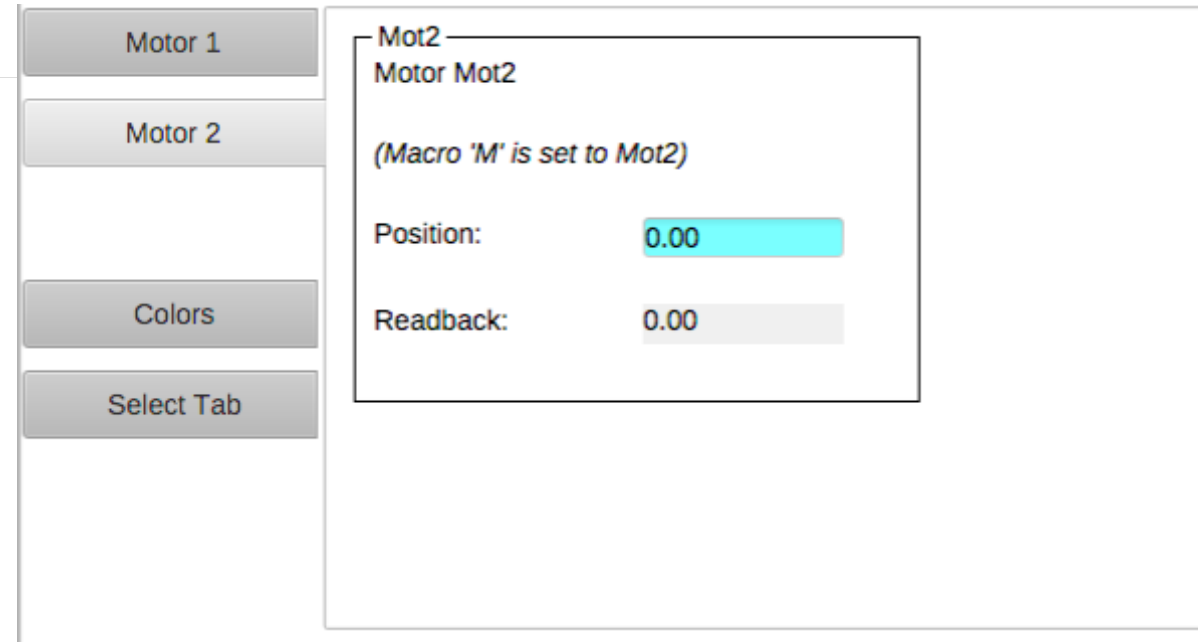
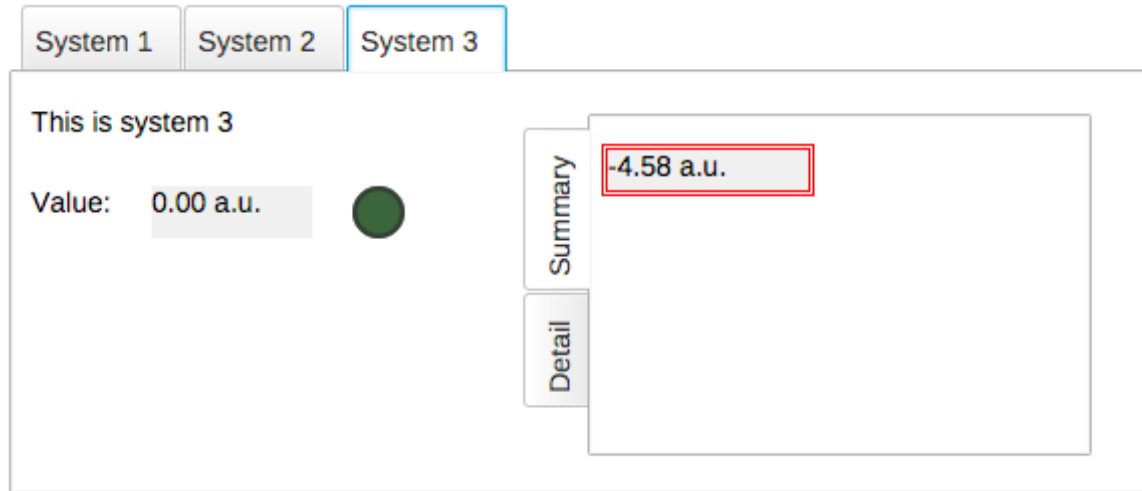
- With “Replace”, can still use

- Context menu



- Control (⌘ on Mac) for tab
- Shift-Control for window

# Screen Navigation: Tabs



## Tabs

Each tab is in-memory, same \*.bob

- Appears immediately when selected
- Uses CPU and memory when hidden

## Navigation Tabs

Tab is loaded from separate \*.bob when selected

- May need a little time to load
- No CPU and memory when hidden

# Macros

- Macros are passed into displays from
  1. Enclosing Group or Tab Widget
  2. Display
  3. Embedded widget container or Action that loaded the display
  4. Phoebus preferences

- To use: `$(NameOfMacro)`

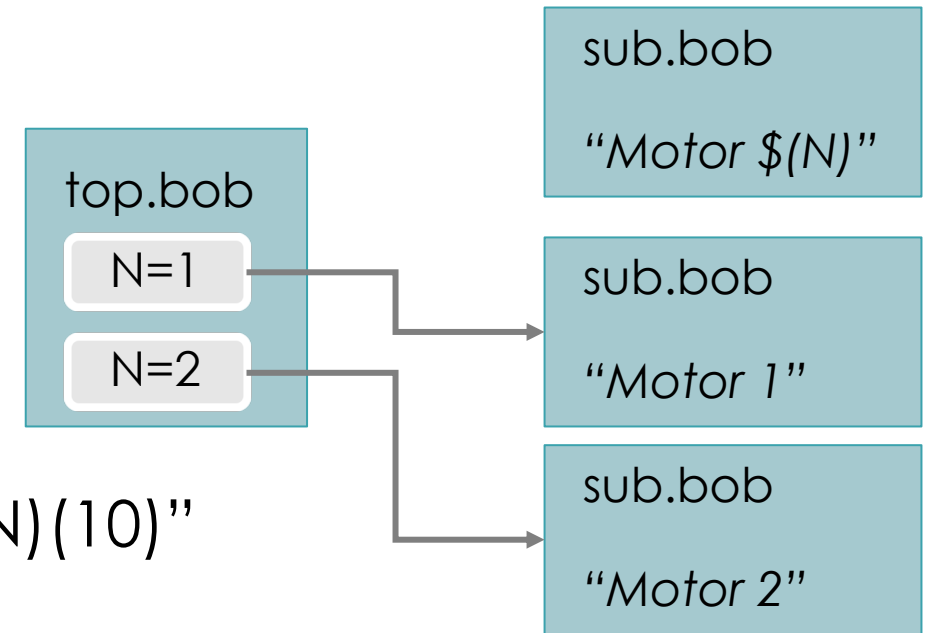
.. or `${NameOfMacro}`.  
EPICS \*.db files use `$(xx)`,  
SNL and shell use `${xx}`,  
so we support both conventions.

- Examples:

PV Name:	<code>\$(PV)</code>	with <code>PV=TheFullPVName</code>
PV Name:	<code>Motor\$(N)</code>	with <code>N=1, 2, 3, ...</code>
Width:	<code>\$(WID)</code>	with <code>WID=200</code>
Visible:	<code>\$(SHOW)</code>	with <code>SHOW=true</code>

# Macro Example

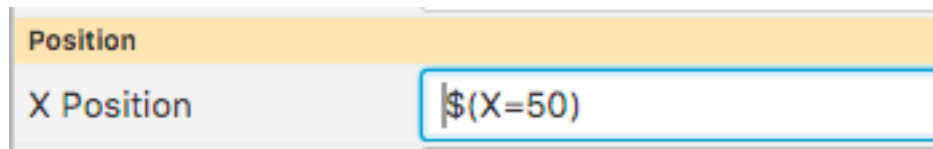
1. Create display “sub.bob”
  - Label with text “Motor \$(N)”
  - TextUpdate with PV “loc://pos\$(N)(10)”
  - ActionButton with PV Name “loc://pos\$(N)(10)” and Action to “Write PV” value 20
  - Copy that button, update to set PV to 30
2. Create display “top.bob”
  - ActionButton with Action to open sub.bob with N=1
  - Copy/paste the button, update to N=2
3. Execute top.bob, press buttons



# Macros

- Default values:  $\$(\text{MACRO}=\text{default})$

Allows standalone testing without passing values into display




Position  
X Position

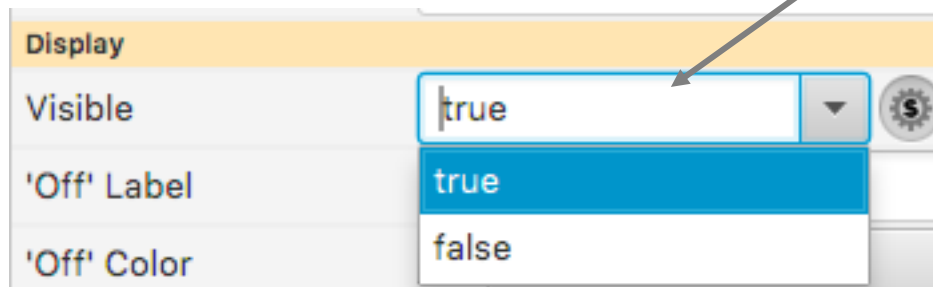
- To enter macro for Boolean  
Press the “\$” macro button


Select valid option from drop-down ...

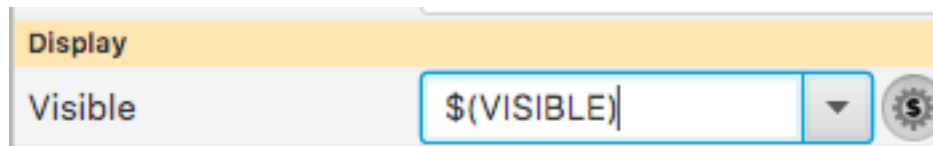



Display  
Visible  

.. or enter a macro



Display  
Visible    
'Off' Label true  
'Off' Color false



Display  
Visible  

# Macro Fallbacks

When macro is not defined, falls back to

- Widget Properties

- Uses the internal property name shown in tool-tip of Properties view
- Note how tooltip is usually preset to “\$(pv\_name)\n\$(pv\_value)”
- Action Button has PV Name property.  
It’s not used directly as in other widgets with PV name, but in “Write PV” the PV name is preset to \$(pv\_name)
- Action Button text is preset to “\$(actions)”

- Java Properties

- \$(os.name)

- Environment Variables

- \$(HOME), \$(USER)



# Predefined Macros

\$(DID): Unique display identifier, useful for per-display PVs

loc://x\$(DID)(10)

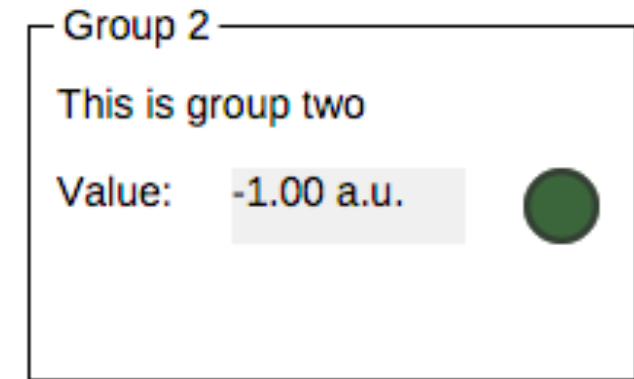
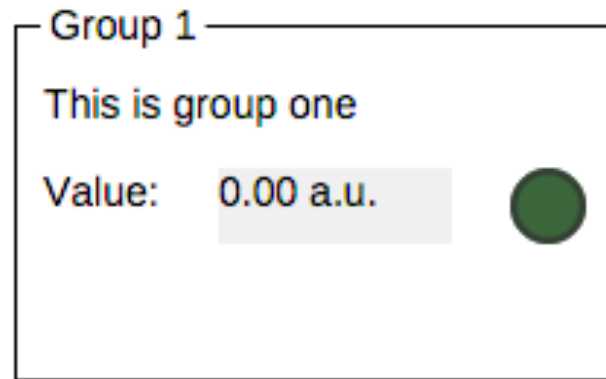
\$(DNAME): Display Name

# Group Widget

Contains other widgets

Visual Effect:

- Border, Name



Practical Effect:

- Group can define macros for contained widgets
- Group can be moved, copied/pasted as one unit in editor

# Group Widget


1) Add Group Widget

2) Move other widgets inside the Group

Active Group is highlighted

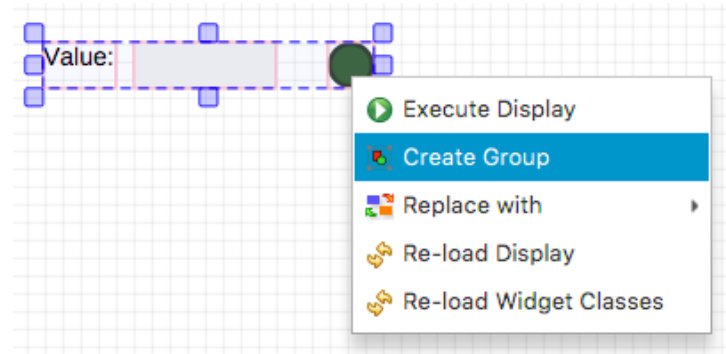
# Group Properties

- Name:  
Shown in border
- Style:  
“Group Box” for named border
- Macros:  
Passed to contained widgets

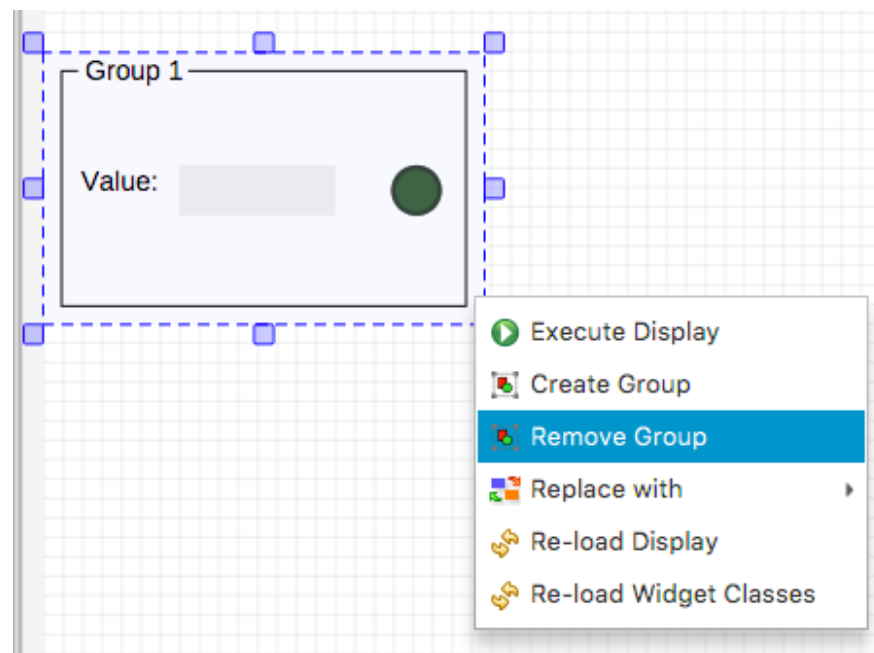
Properties	
<input type="text" value="Search"/>	
<b>Widget</b>	
Type	 Group
Name	<input type="text" value="Group 1"/>
Class	DEFAULT <span>▼</span>
Macros	[PV = 'sim://sine', TE...]
<b>Position</b>	
X Position	<input type="text" value="0"/>
Y Position	<input type="text" value="91"/>
Width	<input type="text" value="227"/>
Height	<input type="text" value="140"/>
<b>Display</b>	
Visible	<input checked="" type="checkbox"/> <span>⚙️</span>
Style	Group Box <span>▼</span> <span>⚙️</span>
Font	Default
Foreground Color	<input type="color" value="black"/> Text
Background Color	<input type="color" value="white"/> Background
Transparent	<input type="checkbox"/> <span>⚙️</span>
<b>Behavior</b>	
Actions	No action
Rules	0 rules
Scripts	0 scripts
Tool tip	<input type="text"/> <span>⋮</span>

# Group Editing Shortcuts

1. Select Widgets
2. Context menu “Create ..”



1. Select Group
2. Context Menu “Remove..”



# Embedded Display

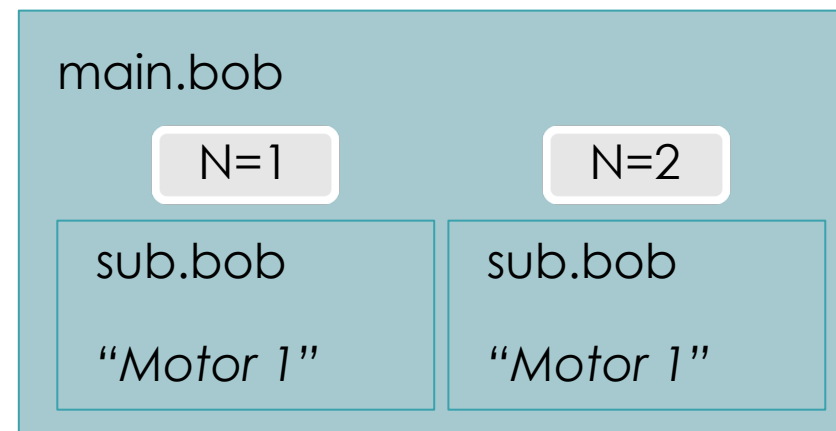
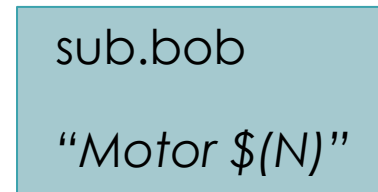
Hosts a complete \*.bob file within a widget

Allows composing higher-level displays from smaller displays:

- Per-device \*.bob
- Show multiple devices in one display

# Embedded Display Example

1. Create display “sub.bob” (or use the one created earlier)
  - Label with text “Motor \$(N)”
  - TextUpdate with PV “loc://pos\$(N)(10)”
2. Create display “main.bob”
  - Embedded Display, File sub.bob, Macros N=1
  - Copy/paste the Embedded Display, update to N=2
3. Execute main.bob





# Embedded Display Sizes

## a) Embedded Display Size

- Size of the widget that will host the \*.bob
- Defined by the widget Width and Height properties

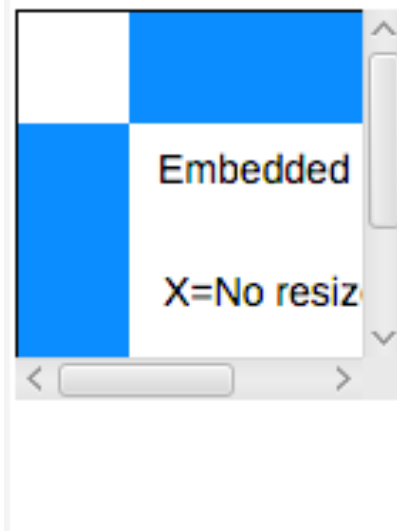
## b) Content Size

- Size of the \*.bob
- Defined by that Display Width and Height properties

What if those sizes differ?

# Embedded Display Resize Options

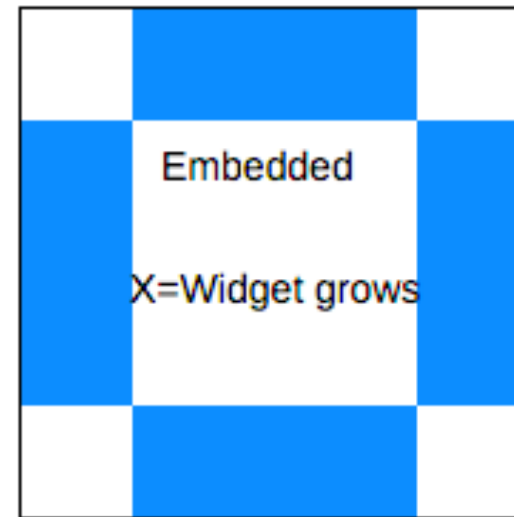
No Resize



Size content  
to fit widget

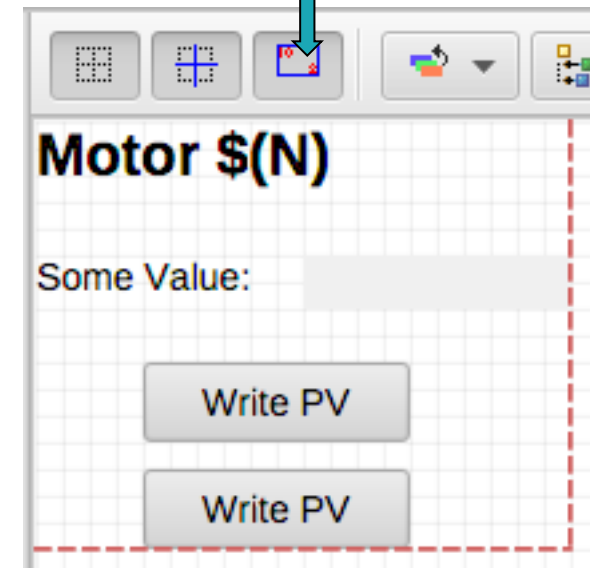
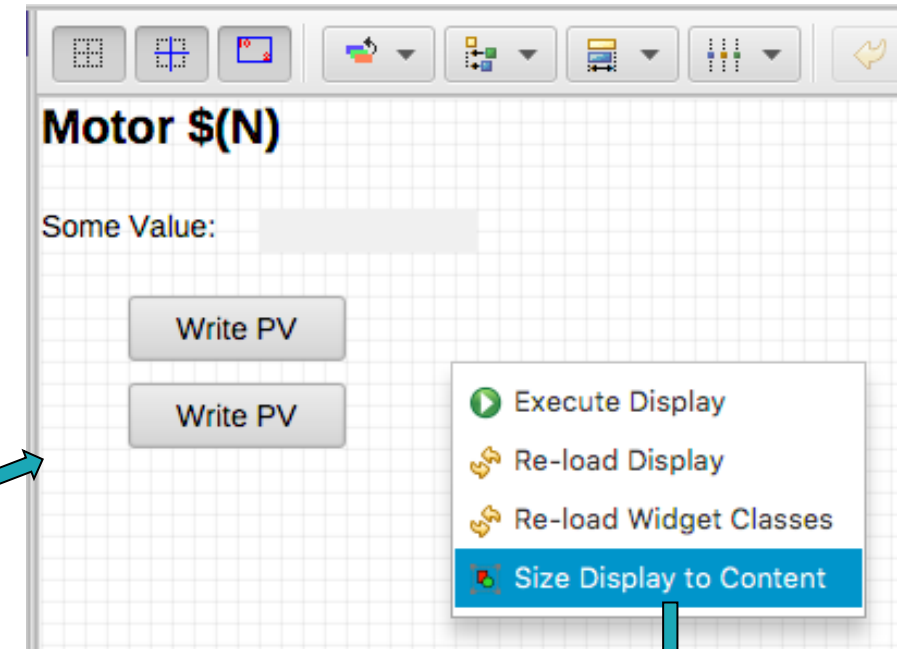
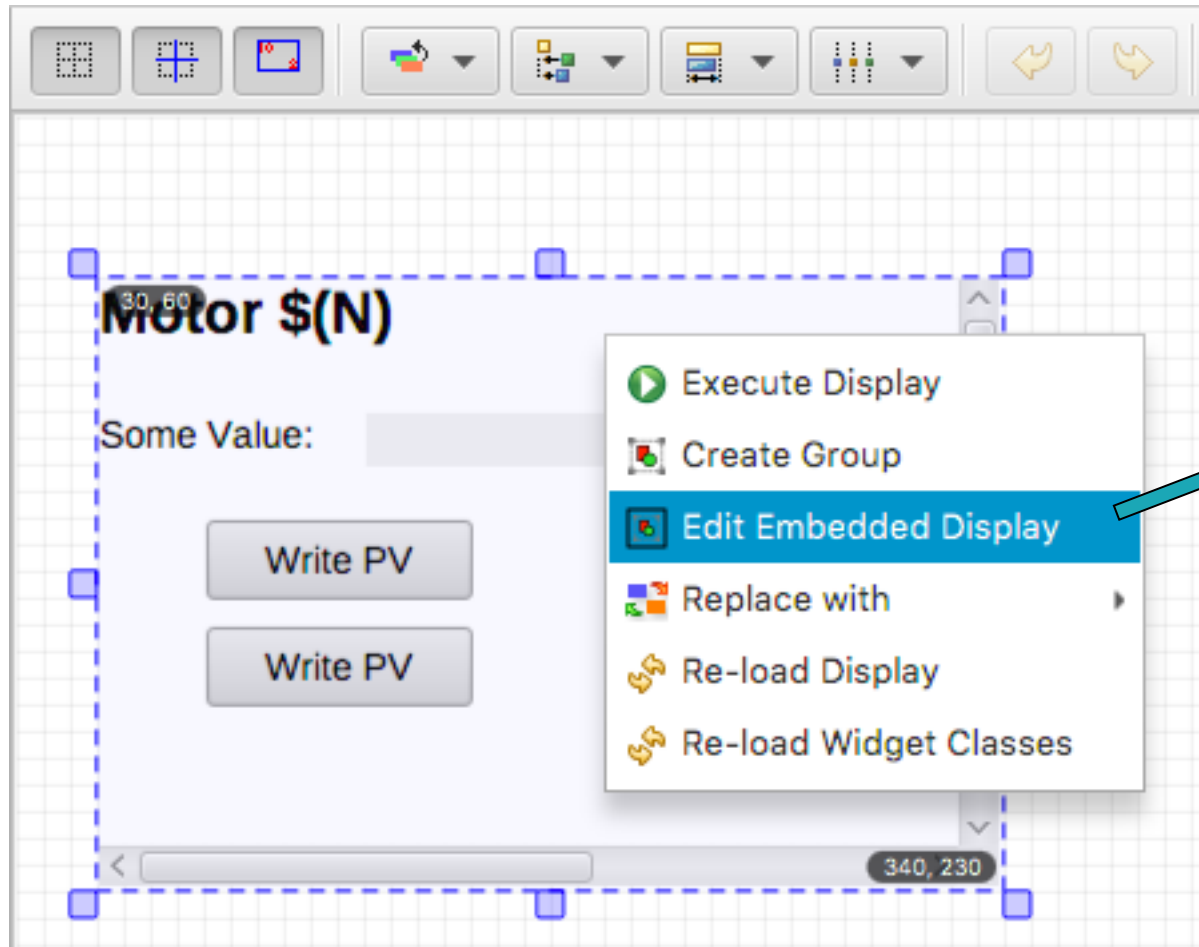


Size widget  
to fit  
content



- ✓ No Resize usually best. Scrollbars appear as needed.
- Resizing results in odd font sizes or widgets that outgrow their initial space.

# Embedded Display Editing



# Top Resources

## See Help, Preference Settings

```
# -----  
# Package org.phoebus.ui  
# -----
```

```
# 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
```

## Start phoebus with “-settings /path/to/my\_settings.ini”:

```
org.phoebus.ui/top_resources=/home/controls/displays/main.bob, Start Page |  
http://controls.my.site/displays/main.bob, Start Page
```

- File system: Use NFS or ‘git pull’ to distribute files
- http: All users always see the same set of files

***TO BE  
CONTINUED...→***

# Many Widgets and Properties

Compared to earlier EPICS display tools,

- **Group Widget** instead of Lines
- **LED** instead of Circle-with-changing-color
- **Tab/Navigation Tabs** instead of buttons, local PVs, conditional visibility,..

Display describes **Meaning**:

- Group of related widgets
- LED for binary PV, not circle that happens to change color

*Files with meaning are easier to translate into the next tool*

# Widget Classes

- Instead of creating a Label with large font, define a “TITLE” class for the Label
- Instead of creating an LED with Orange color, define a “WARNING” LED class

```
# -----  
# 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
```



# Editing \*.bcf Widget Class Files

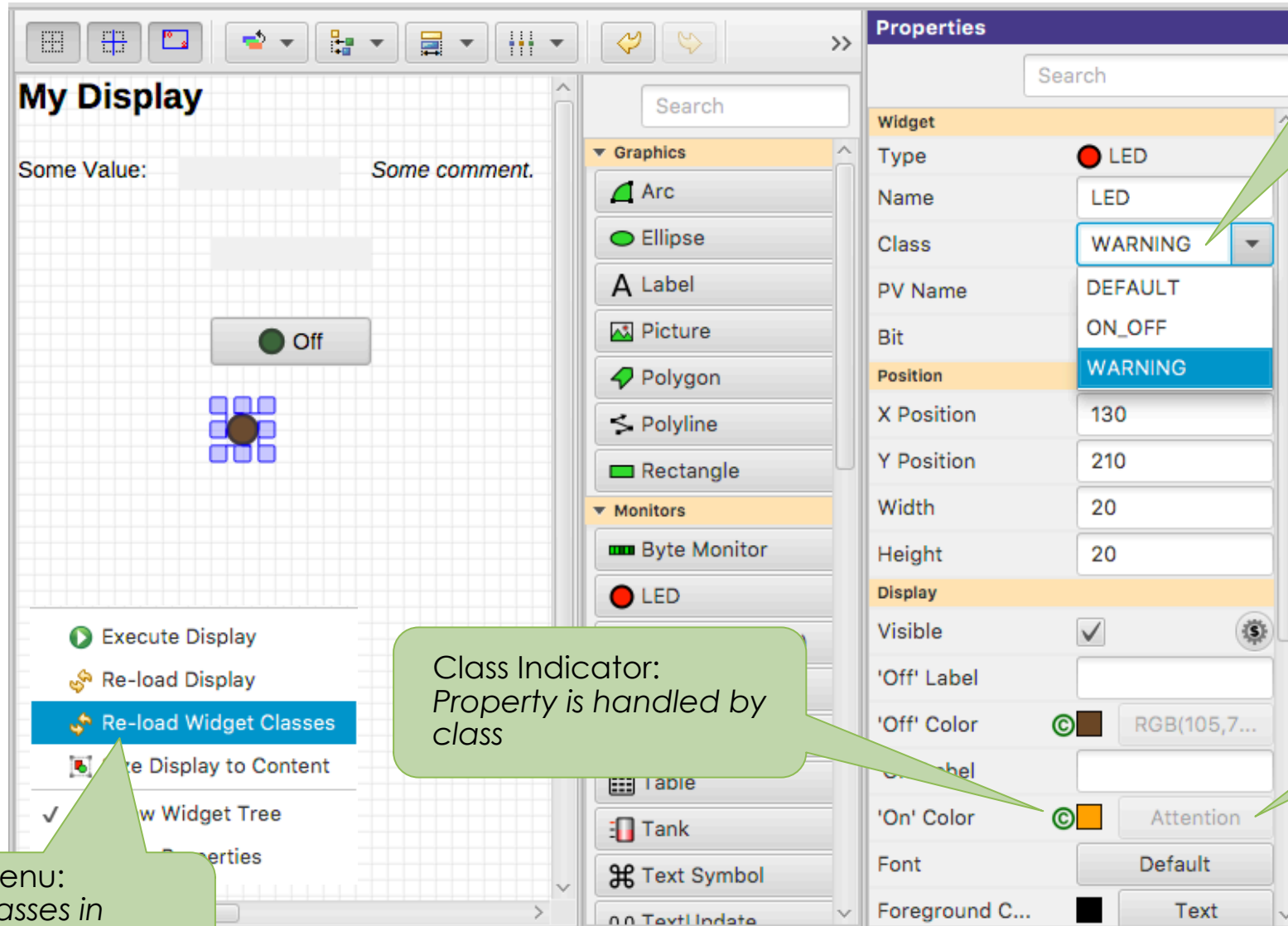
*Slightly different editor behavior*

**Name** Defines a widget Class:  
'WARNING' LED,  
'TITLE' Label,  
...

The screenshot shows the 'Widget Classes' editor. On the left, a 'Widgets' list contains: TITLE, COMMENT, ON\_OFF, WARNING (highlighted), SECTION, and several 'A' entries. The central canvas displays a grid with a 'TITLE' label, a 'COMMENT' label, an 'ON\_OFF' indicator (a green circle), and a 'WARNING' indicator (a brown circle surrounded by a blue grid). The right side features a 'Properties' panel for the selected 'WARNING' widget. The 'Type' is 'LED'. The 'Name' is 'WARNING'. The 'Position' is X: 120, Y: 190, Width: 20, Height: 20. The 'Display' section has 'Visible' checked. The 'Off' color is set to RGB(105,74,44). The 'On' color is set to 'Attention' (checked), with a tooltip that says 'Include this property in definition of widget class?'. The 'Font' is 'Text' and the 'Line Color' is RGB(50,50,50,178). A central toolbar includes icons for grid, zoom, and other editing functions. A 'Monitors' toolbar on the right lists various widget types like Arc, Ellipse, Label, Picture, Polygon, Polyline, Rectangle, Byte Monitor, LED, LED (Multi State), Progress Bar, Symbol, Table, and Tank.

**Checked Property:**  
*Value becomes part  
of class definition*

# Using Widget Classes



Select Widget Class

Class Indicator:  
*Property is handled by class*

Disabled:  
*Cannot change the class-based property*

Context Menu:  
*Re-load classes in case \*.bcf is changed while editing display*

# Class Details

- \*.bcf files define widget classes
  - Label of class *TITLE* uses font XYZ
- When editing a \*.bob file, classes are applied.  
Add Label, select Class *TITLE*:
  - Font is set to XYZ
  - Can no longer change the font
  - File is saved with font=XYZ, marked as “use\_class”
- \*.bob files use widget classes, if they are defined.  
Open a file with Label of class *TITLE*, and
  - a) *TITLE* is a known class:  
Whatever that class defines is used. If it sets font=EFG, that'll be used.
  - b) *TITLE* is not a known class:  
Using font=XYZ as saved in file.

# Compare \*.bcf and \*.bob to \*.css and \*.html

\*.bcf classes are similar to \*.css style settings,  
\*.bob files are similar to \*.html content

a) Have same \*.bcf/\*.css

→ Display looks the same

b) Use different \*.bcf/\*.css

→ Display looks as requested in my \*.bcf/\*.css

c) Have no \*.bcf/\*.css

→ \*.html turns into rubbish, lacking any description of what to look like.

\*.bob display looks as seen by last person who edited it,  
since the class settings effective at that time are in the \*.bob file.

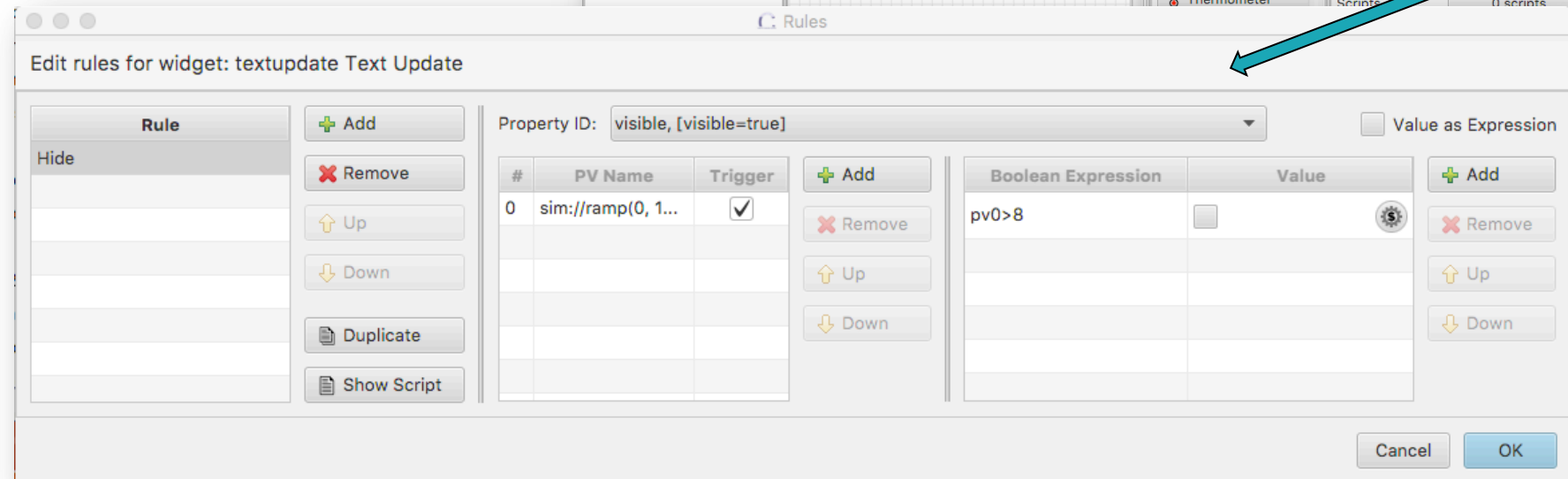
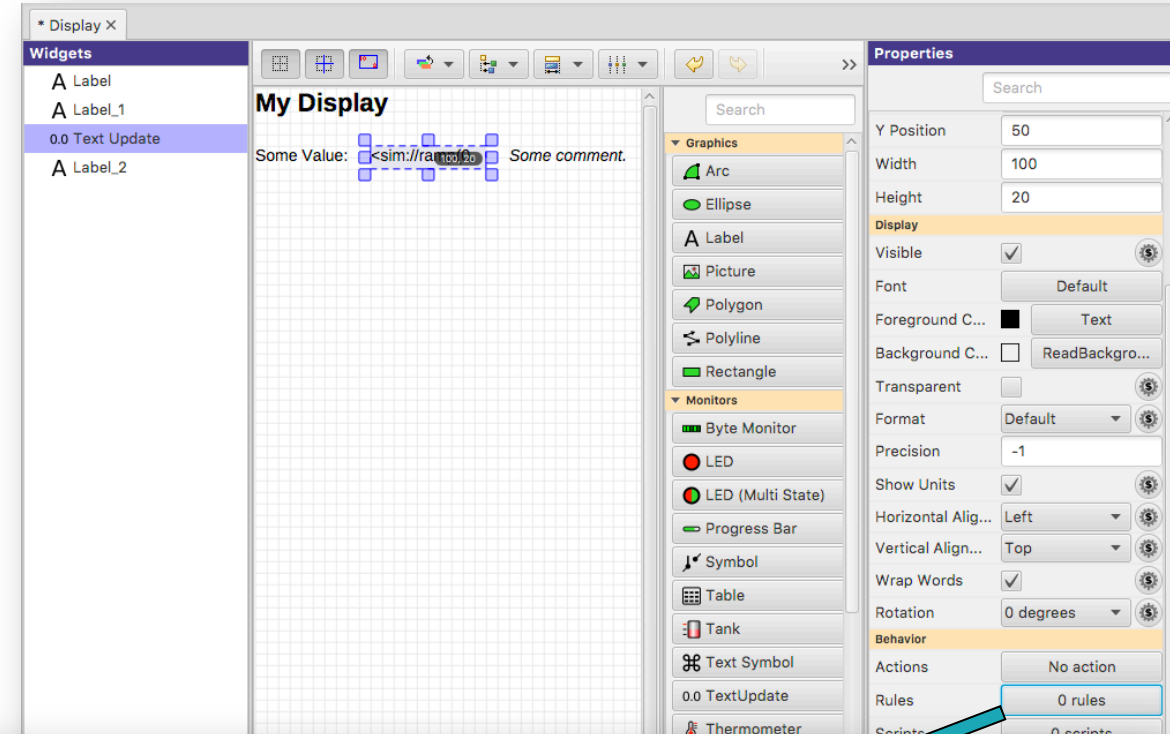
# Rules

- Ideally, use widgets' built-in functionality
  - Value of PV displayed in TextUpdate, LED, ..
  - Alarm indicated via Border
- Sometimes useful to for example hide a widget, i.e. change visibility based on a PV
  - Rules can accomplish this
  - .. But functionality may not be obvious to the next person who needs to maintain a display



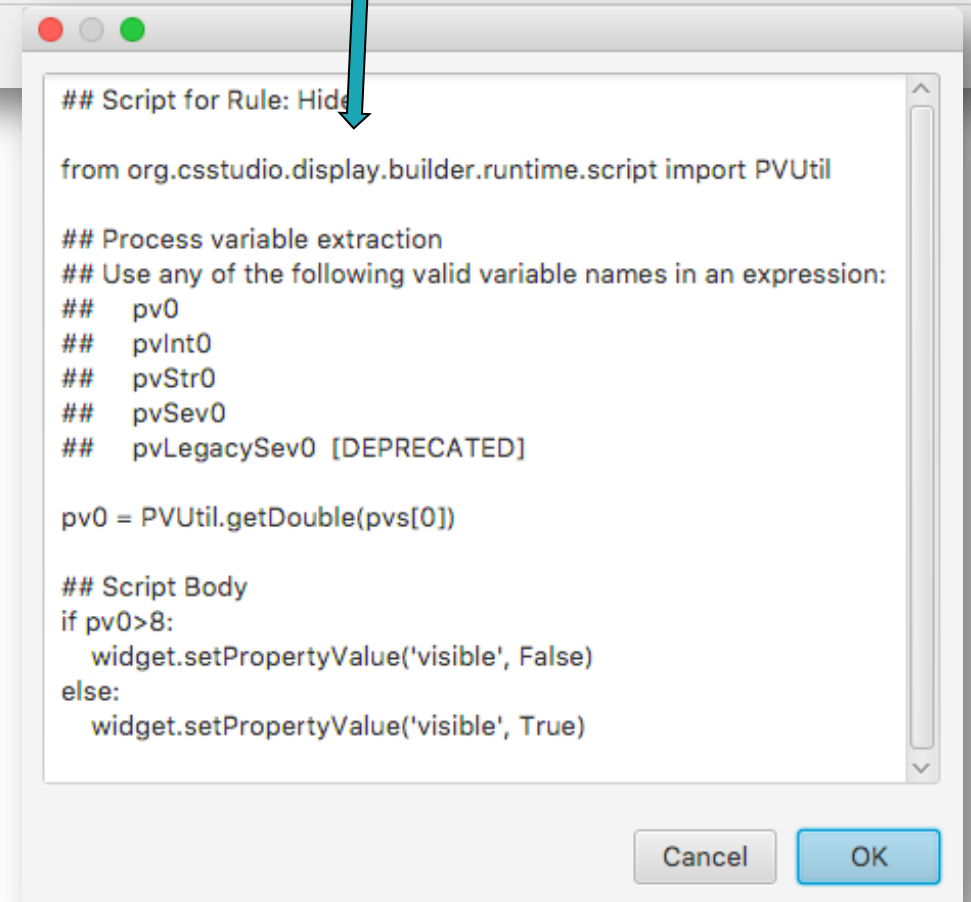
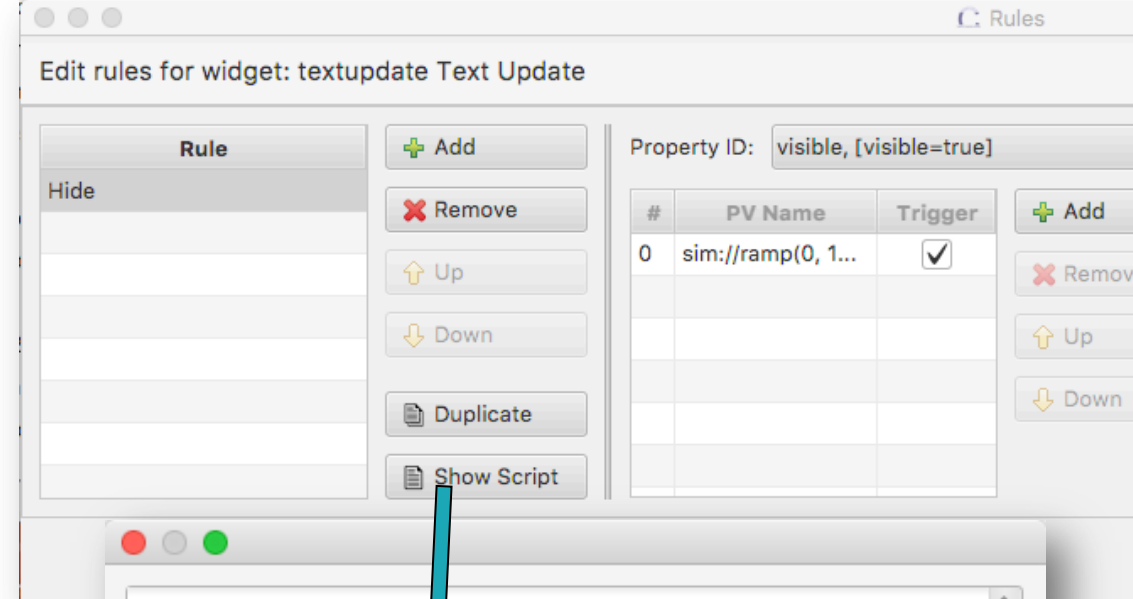
# Adding a Rule

- Add TextUpdate widget
- Set PV to sim://ramp(0, 10, 1)
- Open Widget's Rules
- Add Rule, name it "Hide"
- Select "visible" property
- Add PV sim://ramp(0, 10, 1)
- Add Boolean Expression "pv0>8"
- Un-check value
- Run



# Rules Detail

- Triggered by at least one PV
  - May use additional non-trigger PVs
- Expressions use pv0, pv1, ..., pvStr0, pvStr1, .. to access PVs' values
- Rule internally converted to Jython
  - Use preview to debug
- “else: ..” sets property to original value





# Scripts

- Scripts are attached to a widget
- Triggered by at least one PV
  - May use additional non-trigger PVs
- Invoked with
  - pvs[] – Array of requested PVs
  - widget – The widget
- Script can
  - Read & write the received PVs
  - Set widget properties
  - Locate other widgets in the display
  - Invoke any Java code in the product
  - Be very powerful
  - Result in an unmaintainable mess
- One Script Executor per \*.bob file, Runs in background thread
  - Slow scripts do not block the UI
  - One script per display at a time
    - a) Many short-duration scripts
    - b) One that never quits

# Rules vs. Scripts

- Both are in the end Jython code
- Both should be the exception.  
Plain displays don't need them.  
But can be powerful,  
replacing separate custom Java/Python/C/C++ applications.
- Prefer Rules because they describe meaning, easier to maintain

# When to use a script

- It's simple, well documented, and tremendously improves the UI
- Would be a one-of, specialized, hard to maintain, separate application anyway.  
With a script, at least its integrated into the operator UI

## Examples:

- Turn scalar PVs into `loc://waveform` for guideline in XYPlot
- Fill display with 50 widgets based on config file, `examples/template_and_script`
- Add information from web service to display

# When not to use a script

- It adds logic to the display that should be on the IOC
  - Display should only display PVs and allow user to write PVs.
  - Display must never do anything
- You have to ask for help implementing the script
  - If you can't implement it, you can't maintain it, either

## Examples

- Open relieve valve when pressure too high.  
Ramp Power Supply.
  - What if somebody closes the display? Opens two displays?
- Wiggle something on the display
  - It's not a video game



# Summary

Display Builder is powerful Editor and Runtime with many Widgets, Macros etc.

Keep it Simple

1. Add a Widget
2. Enter Label's Text or Widget's PV Name
3. Done

**Sample & Detector**

	Destination Pos	Current Pos	
SANGLE	0.5000 deg	0.5012 deg	Scan
SampleX	-8.9618	-8.9618	Scan
Beam Stop	0.0362 mm	0.0362 mm	Scan
Sample Changer	Undefined	-87.0008 mm	Scan
DANGLE	13.0000 deg	13.0015 deg	Scan

**Slits - Collimation**

	Destination Pos	Current Pos		Destination Pos	Current Pos		
S1HWidth	0.500 mm	0.501 mm	Scan	S1VHeight	30.000 mm	29.998 mm	Scan
S2HWidth	3.000 mm	2.996 mm	Scan	S2VHeight	30.000 mm	30.010 mm	Scan
S3HWidth	0.500 mm	0.500 mm	Scan	S3VHeight	40.000 mm	40.000 mm	Scan

**Slits - Background**

	Destination Pos	Current Pos	
RSlit4	-58.5160 mm	-58.5165 mm	Scan
BDetSlit	0.0438 mm	0.0455 mm	Scan
RDetSlit	-4.9926 mm	-4.9927 mm	Scan

**Pump Maintenance (RP)**

Current Running (Hours)	Alarm Setpoint (Hours)	Maintenance Require if RED
RP01	25	8000
RP02	26	8000
RP03	25	8000
RP04	26	8000
RP05	128	8000
RP06	127	8000
RP07	3244	8000
RP08	0	8000
RP09	3245	8000
RP10	3245	8000

**Pump Maintenance (TMP)**

Current Running (Hours)	Alarm Setpoint (Hours)
TMP01	2204
TMP02	26
TMP03	26
TMP04	25

**X-Y ROI**

	Min	Max	Mean	Total	Total +/-	Rate
Signal	11	363	116.194	22774	150.911	0 e/s
Background	0	1425	71.161	160824	401.029	0 e/s
S/B	0.000	0.255	1.633	0.142	0.001	0.000

**QIE ROI**

	Min	Max	Mean	Total	Total +/-	Rate
Signal	44	8996	778.156	105051	324.116	0 e/s
Background	6	2090	182.778	24675	157.083	0 e/s
S/B	7.333	4.304	4.257	4.257	0.030	0.000

**Data Collection**

Total Counts	4890418	0 e/s
Proton Charge	4.0063559E+1 C	
Beam Power	1402537 Watts	
Data Collection State	Idle	
Data Collection Pause	Not Paused	

**QIE Axes & ROI Position Details**

	ROI Start	ROI Size	Start	End	Bin Size
Q Signal	4.268	1.067	Q Axis	0.0000	13.9396
E Signal	0.000	7.500	E Axis	-80.0000	80.0000
Q Background	2.134	1.067			
E Background	15.000	7.500			