Hall A High Voltage EPICS Screens
CS-Studio Phoebus

Brian Eng
DSG
2021/09/09
Why Phoebus?

• No longer dependent on Eclipse
  – Standard Widget Toolkit (SWT) is one of the toolkits to create GUIs in Java
  – More than just SWT, Eclipse is an Integrated Development Environment (IDE)
    ▪ Heavyweight (30+ min vs ~3 min compile time)
    ▪ More code, e.g. 11.2 kloc vs 4.5 kloc for Channel Finder

• Since ~2016, various parts of CS-Studio have been migrating to using JavaFX as the GUI toolkit
  – Included with the Java Development Kit (JDK) since Java 11
    ▪ also available separately for older JDKs
  – Phoebus has all components in JavaFX
Why not stay with CSS-BOY?

• Phoebus is more performant: less CPU % and less memory use
  – Varies based on screen, Heater Demo: ~¼ CPU (~40% vs ~10%) and ~½ memory usage (~9 MB vs ~4 MB)

• Phoebus can import and run BOY screens with no changes
  – Major exception are scripts
    ▪ Generally, just need to change import path
    ▪ Highly dependent on the script (one reason to avoid script use)

• Most BOY developers have moved to Phoebus
  – New widgets and features not in BOY
Hall A EPICS Screens

• Hall C recently upgraded from some Tcl/Tk screens to CSS-BOY
  – Scripts to generate CSS-BOY done by DSG (Lemon & Bonneau)
  – Upgrading screens and creating new scripts for Hall A

• Initial focus will be on detector high voltage
Creating Screens

• Phoebus screens are XML files with an extension of .bob (BOY is the same, but with .opi)

• One method of creating screens is with the built-in Display Builder editor
  – The default choice
  – Good when layout won’t change

• Some Hall A screens need to be easily customizable (Example #1) or use external parameter files (Example #2)
  – Using Python to generate .bob files
Python Example #1

- Hall A will have experiments that use a suite of detectors in slightly different configurations.
- For the main menu, easier to generate the screen dynamically from a dictionary.
Python Example #2 (1/4)

• Some high voltage supplies have separate set and read process variable (PVs)

• Each high voltage channel has a calculated PV (Vdiff) that indicates difference between set and read
  – |Vset – Vread|
  – PV alarm limits
    ▪ 25 V for LOW and HIGH
    ▪ 50 V for LOLO and HIHI
• When using the monitor widget and its alarm status, default changes only widget border, based on alarm; request was for entire widget to change color

• vdiff.bob uses a rule to change widget background color based on severity level of the PV passed via macro
Python Example #2 (3/4)

- A mapping file (high voltage name, PV, row, column, etc.) can be used to generate a grid of indicators to show Vdiff status, using Python.

- To make the grid, multiple copies of vdiff.bob are embedded into the new detector screen and macros are used to define the individual PV for each monitor widget.
Python Example #2 (4/4)

• Read the high voltage mapping file with Python to generate a grid of embedded displays to create the detector screen
Conclusion

• Phoebus will be used for generating local displays going forward

• Current scripts will be modified and new scripts and screens will be created
  – Next step will be embedding detector Vdiff screens into an overall screen