PyDM
Python Display Manager
EPICS Collaboration Meeting, June 2018

Hugo Slepicka
PyDM

- Open-Source Python-based Framework for Control System Graphical User Interfaces;

- It provides a system for the drag-and-drop creation of user interfaces using Qt Designer;

- Also allows for the creation of displays driven by Python code;

- Intended to span the range from simple displays without any dynamic behavior, to complex high level applications, with the same set of widgets;
Developers can extend the framework with custom widgets for site specific tasks, and plugins for multiple control systems and data sources;

- Makes building control system UIs more efficient;

- Makes operating control systems easier and faster;

- Bridges the gap between control system displays (e.g. EDM panels) and high level applications (e.g. MATLAB or PyQt applications);
Data Plugins

- PyDM ships with the following Data Plugins:
  - Channel Access (ca://)
    - PyEpics
  - Archiver Appliance (archiver://)

- It can be easily extended to support other Data Plugins and protocols such as:
  - ModBus (modbus://) - See: https://github.com/slaclab/pydm_modbus
Widget Set

**Display**
- Byte Indicator
- Image View
- Label
- Log Display
- Related Display Button
- Scale
- Symbol
- Waveform Table

**Input**
- CheckBox
- Enum Combo Box
- Line Edit
- Push Button
- Shell Command
- Slider
- Spinbox
- Waveform Table

**Plot**
- Time plot
- Scatter Plot
- Waveform Plot

**Container**
- Embedded Display
- Frame
- Tab Widget

**Drawing**
- Line
- Rectangle
- Triangle
- Ellipse
- Circle
- Arc
- Pie
- Chord
- Image
PyDM at SLAC

● Deployed:
  ○ **LSST Camera Interlock** Monitoring and Test Suite;
    ■ Using the ModBus Data plugin;
  ○ UI for the **Skywalker Project** (Automatically delivery of the photon beam to a number of experimental hutches at LCLS);
  ○ **LCLS Lightpath** (UI to Quickly Guide Beam to Experimental End Stations);

● Under Development:
  ○ **Typhon** (Interface Generation for Ophyd Devices);
  ○ Building Operator Displays for **Bunch Length (BLEN)**;
  ○ Building Operator Displays for **Klystron Gallery** and other subsystems;
PyDM Beyond SLAC

- **Max Planck Institute for Nuclear Physics in Heidelberg, Germany**
  Using PyDM for a precision mass spectrometer named Pentatrap;

- **LNLS in Campinas, Brazil**
  Using PyDM for the beamlines at current accelerator (UVX) and PyDM will be the standard display manager for the accelerator and beamlines at Sirius, a 4th-generation Synchrotron Light Source;
PyDM Roadmap

- Support to Widget Actions:
  Act on properties (Visibility, Position, Enable Status, etc...) based on Channel Values;

- PVAccess Data Plugin:
  Support for Normative Types and Structured Data in general;
  New Widgets for NTTable and NTNDArray;

- PVAccess RPC Data Plugin:
  Support for Services through PVAccess;

- Expand support for Stylesheets;

- Increase Test Coverage and Documentation;
Screenshots from LSST Camera Interlock
Screenshots from LSST Camera Interlock

Vaccum PLC Test software

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Step</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>TestPlutoGatewayConf</td>
<td>Check Pluto Gateway configuration registers. Expected:</td>
<td>RUN</td>
<td>Pluto Gateway Config match expected values.</td>
</tr>
<tr>
<td>TestPlutoPLCSPresent</td>
<td>Check Pluto Gateway sees Pluto D45 as node 0.</td>
<td>RUN</td>
<td>Pluto Gateway sees D45 PLC as node 0</td>
</tr>
<tr>
<td>TestChannelsBootDefault</td>
<td>Check if all channels are expected when the PLC is</td>
<td>RUN</td>
<td>Checking boot default values. Do not match. IQ20 should be 0. It is</td>
</tr>
<tr>
<td></td>
<td>powered up</td>
<td></td>
<td>True</td>
</tr>
<tr>
<td>TestPlutoWriteReadback</td>
<td>Test write and r/v Pluto adds</td>
<td>RUN</td>
<td></td>
</tr>
<tr>
<td>TestAnalogScaling</td>
<td>Check the analog input wiring, linearity and scaling</td>
<td>RUN</td>
<td></td>
</tr>
<tr>
<td>TestHV Differences</td>
<td>Test HV Pressure absolute difference calculation in the P</td>
<td>RUN</td>
<td></td>
</tr>
<tr>
<td>TestCVValves</td>
<td>Test CVValves</td>
<td>RUN</td>
<td></td>
</tr>
<tr>
<td>TestValveMonitors</td>
<td>Test TestValveMonitors</td>
<td>RUN</td>
<td></td>
</tr>
<tr>
<td>TestHVStat</td>
<td>Test HVStat permit logic</td>
<td>RUN</td>
<td></td>
</tr>
<tr>
<td>TestHVTurboOnOffLogic</td>
<td>Test TestHVTurboOnOffLogic permit logic</td>
<td>RUN</td>
<td></td>
</tr>
<tr>
<td>TestHVTurboPermitBlock</td>
<td>Test TestHVTurboPermitBlock permit logic</td>
<td>RUN</td>
<td></td>
</tr>
<tr>
<td>TestHVPermitAuto</td>
<td>Test TestHVPermitAuto permit logic</td>
<td>RUN</td>
<td></td>
</tr>
<tr>
<td>TestCVStat</td>
<td>Test CVStat permit logic</td>
<td>RUN</td>
<td></td>
</tr>
<tr>
<td>TestCVTurboOnOffLogic</td>
<td>Test TestCVTurboOnOffLogic permit logic</td>
<td>RUN</td>
<td></td>
</tr>
<tr>
<td>TestCVTurboPermitBlock</td>
<td>Test TestCVTurboPermitBlock permit logic</td>
<td>RUN</td>
<td></td>
</tr>
<tr>
<td>TestCVPermitAuto</td>
<td>Test TestCVPermitAuto permit logic</td>
<td>RUN</td>
<td></td>
</tr>
</tbody>
</table>

Run All
Abort

100%  1 / 1

Failed.
Screenshots from Bunch Length (BLEN)
Screenshots from Skywalker

Alignment Control
Procedure: HOMS  Status: Idle
Start  Pause  Abort

Advanced
Lightpath  Expert

XRT DG3M PIM
Position Delta
Beam X  0.0
Beam Y  0.0

Fiducialize  Align

Goals
HX2 PIM  220.0
XRT DG3M PIM  264.0

Mirrors
FEE M1H  225 urad  240.985  Go to Nominal
FEE M2H  141 urad  141.000  Go to Nominal

11-28 09:35:01 Selecting procedure HOMS
11-28 09:34:58 Loading necessary device information from database
11-28 09:34:58 Selecting imager XRT DG3M PIM
Screenshots from Lightpath
Screenshots from Typhon
Screenshots from Max Planck Institute
Where is PyDM?

Code: https://github.com/slaclab/pydm
Docs: https://slaclab.github.io/pydm
Tutorial: https://slaclab.github.io/pydm-tutorial

Open Chat: https://gitter.im/pydm/General

Install with Anaconda: conda install -c pydm-tag -c conda-forge pydm

Available for:
The only valid measurement of code quality: WTFs/minute