Typhon

Automated Generation of PyDM Displays

Teddy Rendahl

U.S. DEPARTMENT OF ENERGY
Office of Science

SLAC NATIONAL ACCELERATOR LABORATORY

Motivation

SLAC

- PyDM opens the door to a more modern UI
  - Integrated with experimental Python environment
  - Include information from multiple data sources
  - Programmatically generate screens using Python
- Thousands of existing EDM screens to convert or recreate
- Four new LCLS-II instruments with a focus on lowered operations budget
  - Less support from engineers to create and maintain screens
  - Less support from staff scientists to assist with user beamtimes

Design Principles

SLAC

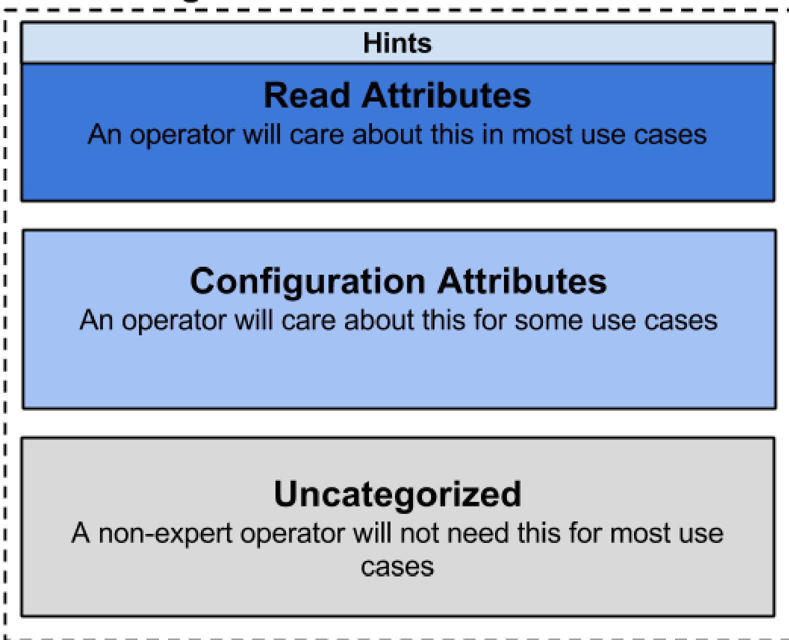
- **Engineering Goals**
  - Avoid ”drag and drop” widget creation wherever possible
  - Decrease the maintenance required to keep screens updated by removing configuration information
  - Provide a clear framework for adding intelligence to screens
- **Operational Improvements**
  - Uniform methodology for displaying complex device structures
  - Importance of control points is clear from screen design
  - Consistency between interfaces lowers operator training time

Typhon

SLAC

- Automatically build PyDM displays based on the structure of an Ophyd Device
- Use the abstraction layer of Ophyd signals with PyDM to display raw and interpreted data
- Consolidate tools formerly kept in separate applications

Ophyd as a Device Data Structure

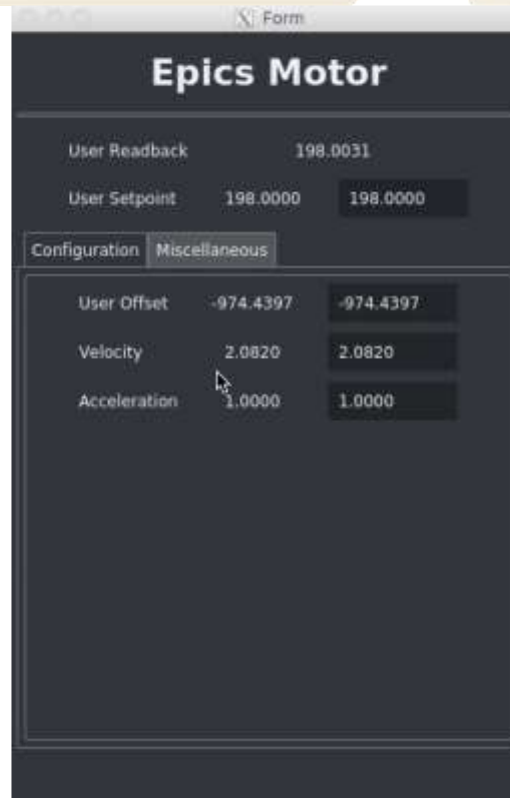


- Contained within an Ophyd Device:
  - Clear component hierarchy
  - Terse but human readable aliases for control points
  - Categorization of signals based on the relevance to standard operation

Basic Signal Organization

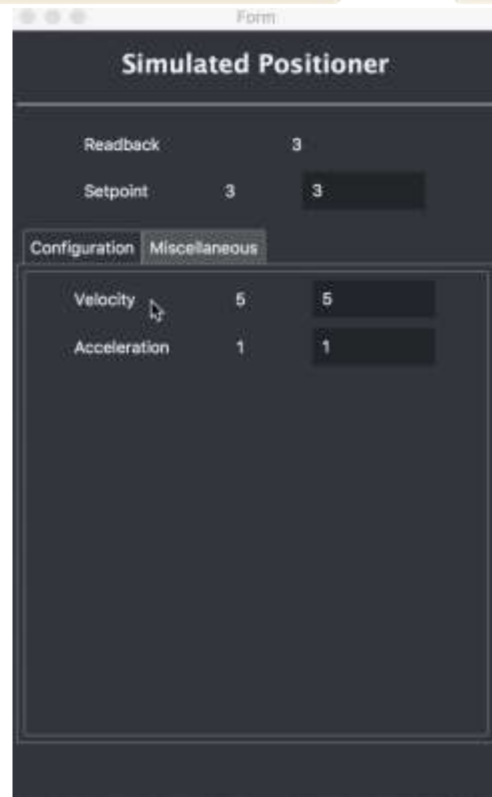
- Show most important attributes at the top while the remainder are into tabs
- Automatically recognize signal types including:
  - Read-Only
  - Enum
  - Simulated

```
1   motor = ophyd.EpicsMotor('Tst:Mtr:01', name='epics_motor')
2   device_display = typhon.DeviceDisplay(motor)
```



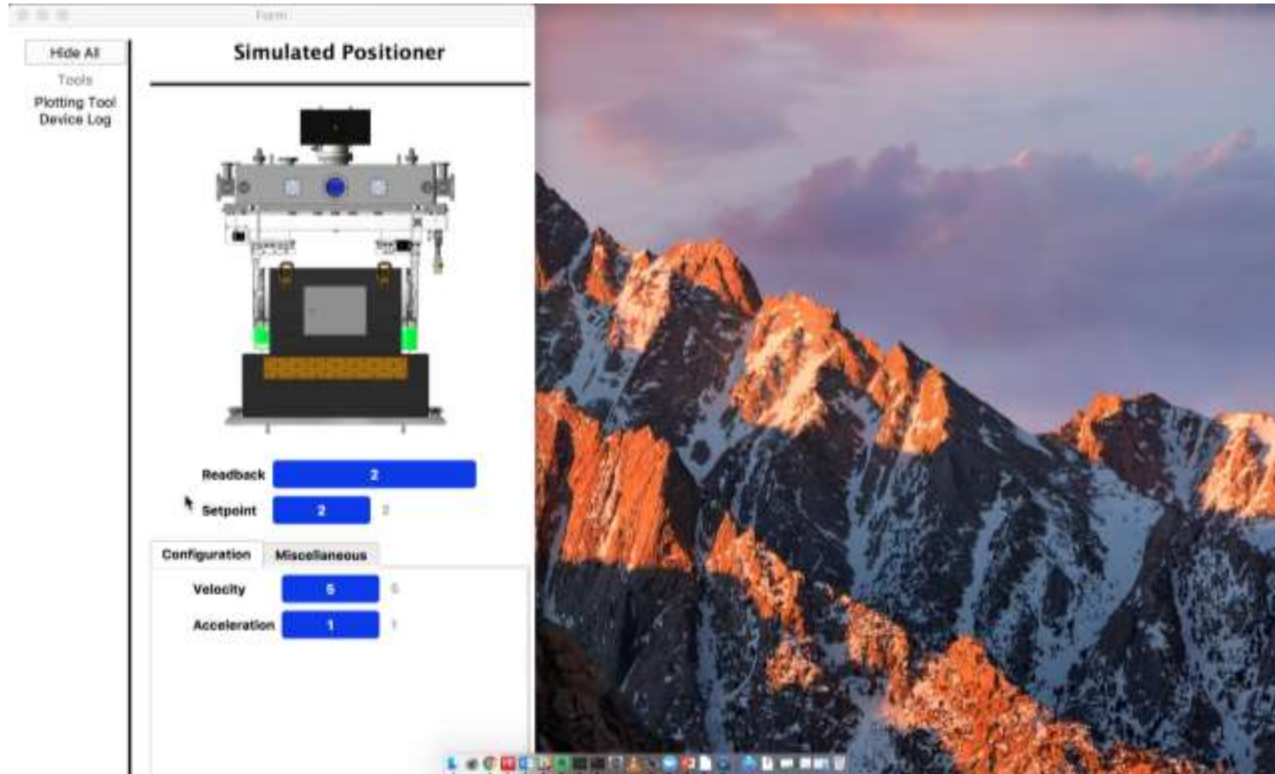
Basic Signal Organization

- Show most important attributes at the top while the remainder are into tabs
- Automatically recognize signal types including:
  - Read-Only
  - Enum
  - Simulated




```
1   motor = SynPositioner(name='simulated_positioner')
2   device_display = typhon.DeviceDisplay(motor)
```

Expanding the Interface



Complex Devices

Hide All

Components

X

Y

Pitch

Tools

Plotting Tool

Device Log

Complex Stage

X Readback 0.000

Y Readback 0.000

Pitch Readback 0.000

Configuration | Miscellaneous

X Velocity 5 5

Y Velocity 5 5

Pitch Velocity 5 5

Including Logic

```
def move(self, x: float, y: float, z: float, simultaneous: bool=True):
    """
    Move the XYZStage
    """
    for stage, request in zip((self.x, self,y, self.z), (x, y, z)):
        stage.move(request, wait=not simulatneous)

def move_radially(self, radius: float, theta: float, phi: float,
                  simulatenous: bool=True):
    """
    Move the XYZStage as specified in spherical coordinates
    """""
    x = r * sin(theta) * cos(phi)
    y = r * sin(theta) * sin(phi)
    z = r * cos(theta)
    self.move(x, y, z, simultaneous=simultaneous)
```

Including Logic

Major Advantages

SLAC

- Common way to present complex devices
- Spend more time thinking about how to organize your device rather than dragging and dropping
- Inclusion of higher level logical operations is simple
- Synergy between command line and Graphical User Interface

Hide All
Components
X
Y
Z
Tools
Plotting Tool
Device Log

Complex Stage

In [13]: xyz.x.velocity = 4

X Readback 0.000
Y Readback 0.000
Z Readback 0.000

Configuration | Miscellaneous

X Velocity 5 5
Y Velocity 5 5
Z Velocity 5 5

On the Horizon

SLAC

- **Current Status**
  - V0.2.0 released in beta
  - Will be deployed in the Fall for operator feedback
- **Planned Improvements**
  - Handle positioners uniquely
  - Inclusion of more external tools
  - Prototype for a RunEngine with fixed plans
  - Beamline process specific screens, not device specific

Support for Complete Screen Lifecycle

SLAC

1. Screen is automatically generated by Typhon
2. Screen is saved to disk for use by operators
3. Adjustments to screens are made by operators
4. As updates to tools and devices are made, screens are re-imported and updated in a programmatic fashion

Links

SLAC

Documentation

https://pcdshub.github.io/typhon/

Repository

https://github.com/pcdshub/typhon