MAPS Qt5/Qt6 GUIs Guide

P. N. Daly

Mountain Operations in Steward Observatory[1]
933 N. Cherry Avenue, Tucson AZ 85719, U S A
(520) 621-3648, pndaly@arizona.edu

Abstract. This document describes the Qt5/Qt6 compliant GUIs associated with the MAPS project on the MMT.

Contents

[1]Steward Observatory is the research arm of the Department of Astronomy at the University of Arizona (UArizona). Its offices are located on the UArizona campus in Tucson, Arizona (US). Established in 1916, the first telescope and building were formally dedicated on April 23, 1923. It now operates, or is a partner in telescopes at five mountain-top locations in Arizona, one in New Mexico, one in Hawaii, and one in Chile. It has provided instruments for three different space telescopes and numerous terrestrial ones. Steward also has one of the few facilities in the world that can cast and figure the very large primary mirrors used in telescopes built in the early 21st century.

List of Figures

1. Installation

1.1. Clone The GitHub Repository

```
git clone https://github.com/pndaly/MapsQt6Guis
cd MapsQt6Guis
```

1.2. Create A Virtual Environment (Optional)

```
python3 -m venv .venv
source .venv/bin/activate
python3 -m pip install –upgrade pip
```

1.3. Install Dependencies

Edit the `requirements.txt` file to select a Qt version, then:

```
python3 -m pip install -r requirements.txt
```

2. Quick Start

In the rest of this document, I assume you will be using Qt5. If you have upgraded to Qt6, replace 'QT_VERSION=5' with 'QT_VERSION=6' in all subsequent commands. To get information on the status or control GUIs use:

```
QT_VERSION=5 python3 maps_status_gui.py –help
QT_VERSION=5 python3 maps_control_gui.py –help
```

The command line arguments are:

HOST the *indiserver* host machine, default='localhost'

PORT the *indiserver* host machine port, default=7624

MODULE the GUI module defined in the TAB_DATA structure in `maps_indi.py`. The choices are *all*, *ao_dm_actuator*, *ao_dm_admin*, *ao_dm_housekeeper*, *ao_dm_operate*, *ao_logger*, *chai2*, *CyberPower*, *Time*, *Tcs*, *Phil*, *Amali* and *New*, default=*all*. Note that this is case-sensitive.

DELAY the event loop delay period in ms, default=2000 (and see 'Timing Loops' in section 7.6. on page 9).

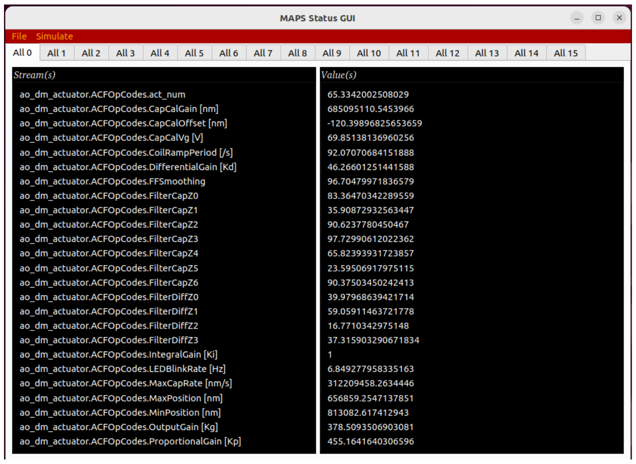
ITEMS the number of INDI streams to display (line-by-line) per page, default=25

FG the foreground colour, default=#FFFFFF (white)

BG the background colour, default=#000000 (black)

To select a specific GUI (for example 'Phil'), just invoke it:

```
QT_VERSION=5 python3 maps_status_gui.py –module=Phil &
```



Stream(s)	Value(s)
ao_dm_actuator.ACFOpCodes.act_num	65.3342002508029
ao_dm_actuator.ACFOpCodes.CapCalGain [nm]	685095110.5453966
ao_dm_actuator.ACFOpCodes.CapCalOffset [nm]	-120.39896825653659
ao_dm_actuator.ACFOpCodes.CapCalVg [V]	69.85138136960256
ao_dm_actuator.ACFOpCodes.CoilRampPeriod [/s]	92.07070684151888
ao_dm_actuator.ACFOpCodes.DifferentialGain [Kd]	46.26601251441588
ao_dm_actuator.ACFOpCodes.FFSmoothing	96.70479971836579
ao_dm_actuator.ACFOpCodes.FilterCapZ0	83.36470342289559
ao_dm_actuator.ACFOpCodes.FilterCapZ1	35.90872932563447
ao_dm_actuator.ACFOpCodes.FilterCapZ2	90.6237780450467
ao_dm_actuator.ACFOpCodes.FilterCapZ3	97.72990612022362
ao_dm_actuator.ACFOpCodes.FilterCapZ4	65.82393931723857
ao_dm_actuator.ACFOpCodes.FilterCapZ5	23.59506917975115
ao_dm_actuator.ACFOpCodes.FilterCapZ6	90.37503450242413
ao_dm_actuator.ACFOpCodes.FilterDiffZ0	39.97968639421714
ao_dm_actuator.ACFOpCodes.FilterDiffZ1	59.05911463721778
ao_dm_actuator.ACFOpCodes.FilterDiffZ2	16.7710342975148
ao_dm_actuator.ACFOpCodes.FilterDiffZ3	37.315903290671834
ao_dm_actuator.ACFOpCodes.IntegralGain [Ki]	1
ao_dm_actuator.ACFOpCodes.LEDBlinkRate [Hz]	6.849277958335163
ao_dm_actuator.ACFOpCodes.MaxCapRate [nm/s]	312209458.2634446
ao_dm_actuator.ACFOpCodes.MaxPosition [nm]	656859.2547137851
ao_dm_actuator.ACFOpCodes.MinPosition [nm]	813082.617412943
ao_dm_actuator.ACFOpCodes.OutputGain [Kg]	378.5093506903081
ao_dm_actuator.ACFOpCodes.ProportionalGain [Kp]	455.1641640306596

Figure 1: The 'all' Status GUI. The LHS shows the INDI stream names or labels and the RHS the current values. The default is 15 items per page so these streams cover 16 tabs. Note the red/orange menubar colouring which indicates that simulation mode is enabled.

3. Status GUIs

The driving principle here is that *anyone* can create their own customized status GUI which does not allow *any* control of INDI streams. Therefore, it is safe to do so. It follows that this is the code that can build *any* GUI it knows about as defined in `maps_indi.py`. After following the steps outlined above, the GUI will be visible and can be run (here I choose the 'all' module as shown in figure 1 on page 4):

QT_VERSION=5 python3 maps_status_gui.py –module=all &

3.1. Running Multiple Status GUIs

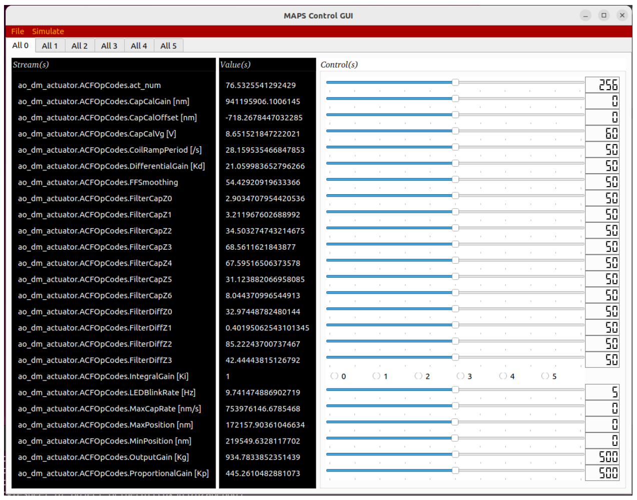
QT_VERSION=5 python3 maps_status_gui.py –module=Phil –items=40 &
QT_VERSION=5 python3 maps_status_gui.py –module=Time &
QT_VERSION=5 python3 maps_status_gui.py –module=CyberPower &
QT_VERSION=5 python3 maps_status_gui.py –module=Amali &

4. Control GUIs

The driving principle here is that there should only be 1 point of control (but this is not currently enforced and it is left to a future developer to add singleton class support). The method to add a new control GUI is exactly the same as adding a new status GUI but the code will only create widgets whose INDI streams are identified as 'wo' (write-only) or 'rw' (read-write). If, for example, you try to run the 'Phil' control GUI, it will not produce any GUI control window but will return:

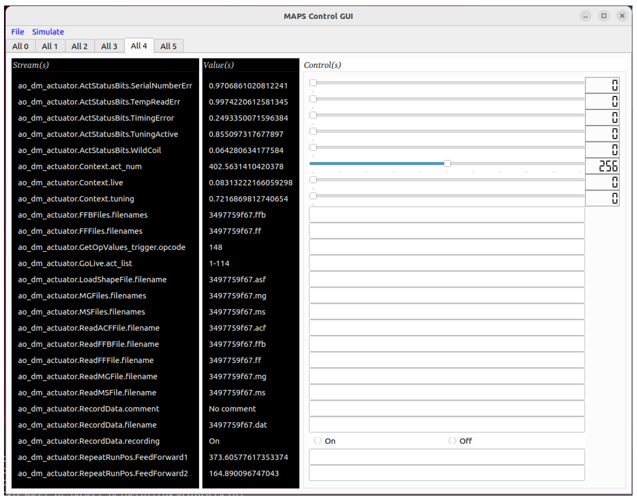
QT_VERSION=5 python3 maps_control_gui.py –module=Phil
2024-07-15 15:40:01,288 line:133 No (writeable) controls selected

This is because all elements within the (pre-defined) 'Phil' GUI are read only so not controllable. However, the following command produces the output(s) shown in figures 2, 3 and 4 on



Stream(s)	Value(s)	Control(s)
ao_dm_actuator.ACFOpCodes.act_num	76.5325541292429	256
ao_dm_actuator.ACFOpCodes.CapCalGain [nm]	941195906.1006145	0
ao_dm_actuator.ACFOpCodes.CapCalOffset [nm]	-718.2678447032285	0
ao_dm_actuator.ACFOpCodes.CapCalVg [V]	8.651521847222021	60
ao_dm_actuator.ACFOpCodes.CoilRampPeriod [/s]	28.159535466847853	50
ao_dm_actuator.ACFOpCodes.DifferentialGain [Kd]	21.059983652796266	50
ao_dm_actuator.ACFOpCodes.FFSmoothing	54.42920919633366	50
ao_dm_actuator.ACFOpCodes.FilterCapZ0	2.9034707954420536	50
ao_dm_actuator.ACFOpCodes.FilterCapZ1	3.211967602688992	50
ao_dm_actuator.ACFOpCodes.FilterCapZ2	34.503274743214675	50
ao_dm_actuator.ACFOpCodes.FilterCapZ3	68.5611621843877	50
ao_dm_actuator.ACFOpCodes.FilterCapZ4	67.59516506373578	50
ao_dm_actuator.ACFOpCodes.FilterCapZ5	31.123882066958085	50
ao_dm_actuator.ACFOpCodes.FilterCapZ6	8.044370996544913	50
ao_dm_actuator.ACFOpCodes.FilterDiffZ0	32.97448782480144	50
ao_dm_actuator.ACFOpCodes.FilterDiffZ1	0.40195062543101345	50
ao_dm_actuator.ACFOpCodes.FilterDiffZ2	85.22243700737467	50
ao_dm_actuator.ACFOpCodes.FilterDiffZ3	42.44443815126792	50
ao_dm_actuator.ACFOpCodes.IntegralGain [Ki]	1	○ 0 ○ 1 ○ 2 ○ 3 ○ 4 ○ 5
ao_dm_actuator.ACFOpCodes.LEDBlinkRate [Hz]	9.741474886902719	5
ao_dm_actuator.ACFOpCodes.MaxCapRate [nm/s]	753976146.6785468	0
ao_dm_actuator.ACFOpCodes.MaxPosition [nm]	172157.90361046634	0
ao_dm_actuator.ACFOpCodes.MinPosition [nm]	219549.6328117702	0
ao_dm_actuator.ACFOpCodes.OutputGain [Kg]	934.7833852351439	500
ao_dm_actuator.ACFOpCodes.ProportionalGain [Kp]	445.2610482881073	500

Figure 2: The 'all' Control GUI. The LHSP shows the INDI stream names or labels and the CP the current values. The default is 15 items per page so these streams cover 6 tabs since only a subset are write controllable. Note the red/orange menubar colouring which indicates that simulation mode is enabled. The RHSP contains control widgets scaled to the datarange defined in `maps_indi.py` elminating the need for error checking any inputs.



Stream(s)	Value(s)	Control(s)
ao_dm_actuator.ActStatusBits.SerialNumberErr	0.9706861020812241	0
ao_dm_actuator.ActStatusBits.TempReadErr	0.9974220612581345	0
ao_dm_actuator.ActStatusBits.TimingError	0.2493350071596384	0
ao_dm_actuator.ActStatusBits.TuningActive	0.855097317677897	0
ao_dm_actuator.ActStatusBits.WildCoil	0.064280634177584	0
ao_dm_actuator.Context.act_num	402.5631410420378	256
ao_dm_actuator.Context.live	0.0831322166059298	0
ao_dm_actuator.Context.tuning	0.7216869812740654	0
ao_dm_actuator.FFBFiles.filenames	3497759f67.ffb	
ao_dm_actuator.FFFiles.filenames	3497759f67.ff	
ao_dm_actuator.GetOpValues_trigger.opcode	148	
ao_dm_actuator.GoLive.act_list	1-114	
ao_dm_actuator.LoadShapeFile.filename	3497759f67.asf	
ao_dm_actuator.MGFiles.filenames	3497759f67.mg	
ao_dm_actuator.MSFiles.filenames	3497759f67.ms	
ao_dm_actuator.ReadACFFile.filename	3497759f67.acf	
ao_dm_actuator.ReadFFBFile.filename	3497759f67.ffb	
ao_dm_actuator.ReadFFFile.filename	3497759f67.ff	
ao_dm_actuator.ReadMGFile.filename	3497759f67.mg	
ao_dm_actuator.ReadMSFile.filename	3497759f67.ms	
ao_dm_actuator.RecordData.comment	No comment	
ao_dm_actuator.RecordData.filename	3497759f67.dat	
ao_dm_actuator.RecordData.recording	On	○ On ○ Off
ao_dm_actuator.RepeatRunPos.FeedForward1	373.60577617353374	
ao_dm_actuator.RepeatRunPos.FeedForward2	164.890096747043	

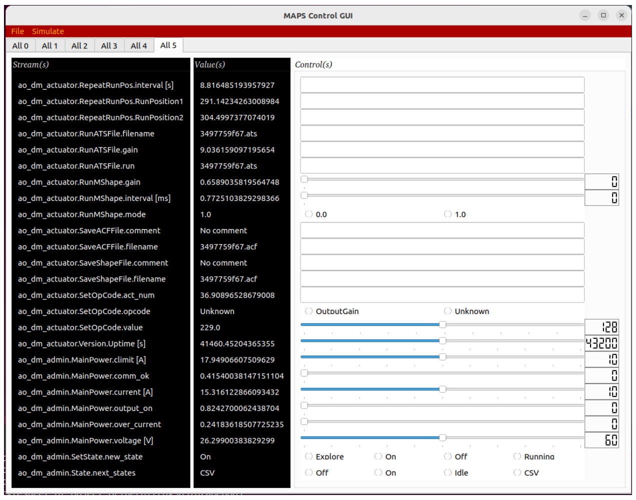
Figure 3: The 'all' Control GUI. This shows a different tab in the same control GUI as above but the pale-green/blue menubar colouring which indicates that simulation mode is disabled.

pages 5, 5 and 6 respectively:

QT_VERSION=5 python3 maps_control_gui.py –module=all

The design principle for *MapsQt6Guis* control has 3 side-by-side panels: the left-hand-side panel (LHSP) shows the data stream name, the central panel (CP) shows the current value and the right-hand-side panel (RHSP) contains the controls for submitting new values. Note that the LHSP and CP are just a re-creation of the status GUI.

Note that the RHSP controls are preferable to the Maga-AO way of doing things which has a single entry widget beneath the current value(s) list because:



Stream(s)	Value(s)
ao_dm_actuator.RepeatRunPos.interval [s]	8.816485193957927
ao_dm_actuator.RepeatRunPos.RunPosition1	291.14234263008984
ao_dm_actuator.RepeatRunPos.RunPosition2	304.4997377074019
ao_dm_actuator.RunATSFile.filename	3497759f67.ats
ao_dm_actuator.RunATSFile.gain	9.036159097195654
ao_dm_actuator.RunATSFile.run	3497759f67.ats
ao_dm_actuator.RunMShape.gain	0.6589035819564748
ao_dm_actuator.RunMShape.interval [ms]	0.7725103829298366
ao_dm_actuator.RunMShape.mode	1.0
ao_dm_actuator.SaveACFFile.comment	No comment
ao_dm_actuator.SaveACFFile.filename	3497759f67.acf
ao_dm_actuator.SaveShapeFile.comment	No comment
ao_dm_actuator.SaveShapeFile.filename	3497759f67.acf
ao_dm_actuator.SetOpCode.act_num	36.90896528679008
ao_dm_actuator.SetOpCode.opcode	Unknown
ao_dm_actuator.SetOpCode.value	229.0
ao_dm_actuator.Version.Uptime [s]	41460.45204365355
ao_dm_admin.MainPower.climit [A]	17.94906607509629
ao_dm_admin.MainPower.comm_ok	0.41540038147151104
ao_dm_admin.MainPower.current [A]	15.316122866093432
ao_dm_admin.MainPower.output_on	0.8242700062438704
ao_dm_admin.MainPower.over_current	0.24183618507725235
ao_dm_admin.MainPower.voltage [V]	26.29900383829299
ao_dm_admin.SetState.new_state	On
ao_dm_admin.State.next_states	CSV

Figure 4: The ‘all’ Control GUI. This shows a different tab in the same control GUI as above which displays entry widets as well as sliders and radio buttons.

1. there is no risk of a typing error writing a valid value to an invalid stream (which could be dangerous);
2. the control type and data limits are taken from the appropriate data structure in `maps_indi.py` so out-of-range values are avoided *i.e. viz.*, there is no risk of an invalid value being written to a valid stream;
3. thee utilization of datatype and datarange elements from the appropriate data structure element in `maps_indi.py`, means that no error checking is required saving lots of lines of code.

5. Adding New GUIs

The principal file for driving new GUI development is `maps_indi.py`. This contains a set of data structures defining the INDI streams associated with each sub-system. I have used the original Elwood Downey names but these are just placeholders so can be renamed in future without loss of functionality. To add a new GUI, use the PND_GUI as an exemplar. In `maps_indi.py`:

1. Create the PND_GUI structure;
2. Add an entry to TAB_NAMES providing a unique key (‘Phil’) and preferred tab name (“Phil’s Shiny New GUI”);
3. Add an entry to TAB_DATA with the same key as the previous step (‘Phil’) and referencing the PND_GUI data structure.

That’s it!

6. Adding New INDI Streams

Adding new streams is as simple as defining a new data structure and including it in the flattened dictionary called ALL_STREAMS in `maps_indi.py`. For example, let us assume we have a new INDI stream called ‘telescope’ and it has 3 ad-hoc data points:

telescope.focus.value this is 'rw' so can be changed within the (known) range -275 through +290 microns;

telescope.truss.temperature this is 'ro' so cannot be changed;

telescope.primary.fan this is 'wo' and can only be toggled between 2 positions 'Off' and 'On'.

So, in the new dictionary each INDI stream is a separate key that contains a sub-dictionary with the following keys:

actval this is the actual value shown by the widget and can be received from INDI or simulated;

datarange this is an appropriate data range. For floats it is typically a bounded tuple. For integers it can be a bounded tuple or a list. For strings, it can either be a string or a list of strings;

datatype the data type. Supported datatypes are 'float', 'int' and 'str'. There is a placeholder type called 'binary' but this is not currently used (see section 7.5. on page 9);

label if specified, the label is used but if it is 'None' a label is populated from the dictionary key;

permission 'ro' for read-only, 'rw' for read-write or 'wo' for write-only;

simval the simulation value or function to be used to calculate one;

tooltip a human-readable pop-up textual hint seen by hovering over the widget;

unit the unit of the data, if there is one;

widget defined as None and populated according to the datatype: 'float' and 'int' can generate sliders, 'int' and 'str' can generate radio buttons, 'str' can generate entry widgets (on the fly).

So, the new dictionary in `maps_indi.py` would look like this:

```
TELESCOPE = {
 "telescope.focus.value": {
    "actval": math.nan,
    "datarange": (-275.0, 290.0),
    "datatype": "float",
    "label": "Telescope Focus",
    "permission": "rw",
    "simval": random.uniform,
    "tooltip": "Telescope focus via the Galil controller",
    "unit": "micron",
    "widget": None,
 },
 "telescope.truss.temperature": {
    "actval": math.nan,
    "datarange": (-25.0, 75.0),
    "datatype": "float",
    "label": None,
    "permission": "rw",
    "simval": random.uniform,
    "tooltip": "Telescope truss temperature",
    "unit": "Celsius",
    "widget": None,
 },
 "telescope.primary.fan": {
    "actval": "Off",
    "datarange": ["On", "Off"],
    "datatype": "str",
    "label": None,
    "permission": "wo",
    "simval": random.choice,
    "tooltip": "Toggle the primary air-blowing fan off or on",
    "unit": "",
    "widget": None,
 },
}
```

We then replace:

ALL_STREAMS = {**{}, **AO_DM_ACTUATOR, **AO_DM_ADMIN, **AO_DM_HOUSEKEEPER, **AO_OPERATE, **AO_LOGGER, **CHAI2, **CYBER_POWER, **TIME, **TCS}

with

ALL_STREAMS = {**{}, **AO_DM_ACTUATOR, **AO_DM_ADMIN, **AO_DM_HOUSEKEEPER, **AO_OPERATE, **AO_LOGGER, **CHAI2, **CYBER_POWER, **TIME, **TCS, **TELESCOPE}

When we edit the `maps_indi.py` file we should always check for data consistency by executing the command:

python3 maps_indi.py

If it returns an error, please correct it before proceeding with any other development. If it does not return an error, we are done and free to use the new INDI streams in the GUIs.

7. Other Considerations

7.1. A Generic Solution

As can be seen, the solution provided is totally generic and relies only on the data within the various dictionaries in `maps_indi.py` being reviewed and kept up-to-date and accurate.

7.2. Simulation Mode

Note that there is a simulation mode enabled at this time to allow software developers or curious users to run the code without the *indiserver* being present. This facility can be toggled on and off but you should *not* assume that turning simulation off automatically connects the code to the *indiserver*. It does not. It is also possible that this simulation mode behaviour will be changed in a future release (including its removal when its utility is no longer valid).

So, to fully connect to the *indiserver*, turn ‘simulate’ off and then use the file menu ‘Connect’ item as shown in figure 7 on page 10. The keyboard shortcut is ‘Ctrl+C’.

Finally, note that if the connection to the *indiserver* fails, the GUI will *automatically* re-enable simulation mode!

7.3. Log Files

Log files are created in `/tmp/maps_status_gui.log` and `/tmp/maps_control_gui.log`.

7.4. Control GUI

The control GUI should be a *singleton* class and, currently, it is not.

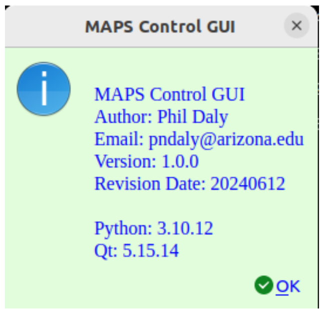
7.5. Binary Large Objects and Redis

At the present time, the code does not handle binary large objects (BLOBs) nor does it handle *redis* publish-subscribe data streams. The former needs thinking about but the latter should be pretty straightforward given the infrastructure code developed here.

7.6. Timing Loops

INDI is a publish-subscribe paradigm mechanism and as such relies on ‘asynchronous i/o’. GUIs, in order to remain responsive to mouse clicks and the like, use the ‘event driven’ paradigm. These two are not necessarily compatible so may not play well together: there is the possibility of a system lock and/or race condition. In the code, I have provided the ‘–delay’ command line argument to adjust one of these loops so that responsiveness and timely updates can peacefully coexist. This delay may well be different for the different GUI modules defined and can only be determined empirically.

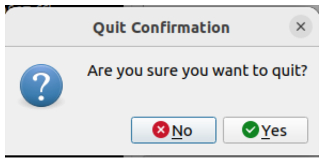
A Miscellaneous Screenshots



MAPS Control GUI
Author: Phil Daly
Email: pndaly@arizona.edu
Version: 1.0.0
Revision Date: 20240612

Python: 3.10.12
Qt: 5.15.14

Figure 5: Rudimentary information on the software is included in the ‘About’ box.



Are you sure you want to quit?

Figure 6: When you exit the GUI, you will be prompted to confirm your decision.

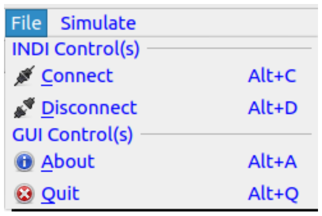


Figure 7: The ‘Filemenu’ contains connect and disconnect buttons plus the ‘About’ and ‘Quit’ controls.