

Python in midas

Ben Smith Midas Workshop 2023



Why python?

- Requested by users at last workshop
- Initial concept was to replace hacky scripts
 - python scripts that called odbedit
 - midas file > mdump > parse text > python analysis
 - many more examples of calling midas command-line tools from Tcl/bash/csh/python/perl scripts
- Python is common (only?) language that students know
 Python interfaces nicely with C

Design goals

- Usability
 - Make it "pythonic"
 - E.g. error-checking via exceptions, not return codes
 - Make simple interfaces with sensible defaults
 - Give people the tools they need
- Maintainability
 - Don't be a burden on the core C/C++ code
 - Make it simple to add new features to python code

What is implemented

- Midas file reader
 - Pure-python implementation
- Midas client
 - ODB access, run transitions, event buffers, RPC, alarms, messages, history...
 - Python wrapper calls the C++ code
- Frontend framework
 - Periodic/polled equipment same concepts as C/C++ frontends, just written in python

File reader

```
import midas.file_reader
```

```
# Get ODB as a dict
odb = my_file.get_bor_odb_dump().data
run_number = odb["Runinfo"]["Run number"]
```

```
# Loop over events in file
for event in my_file:
    # Bank data is either numpy array or python tuple
    some_counter = event.banks["SCAN"].data[0]
```

Midas client

import midas.client

```
if __name__ == "__main__":
    client = midas.client.MidasClient("pytest")

    # Get data from ODB
    state = client.odb_get("/Runinfo/State")

    if state == midas.STATE_STOPPED:
        # Set data in ODB
        client.odb_set("/pyexample", {"an_int": 1, "a_dbl": 4.56})
```

```
# Write message to midas log (set is_error=True for error)
client.msg("Hello from python")
```

Full list of midas.client functions

- ♥ communicate
- ♀ create_alarm_class
- ♀ create_evaluated_alarm
- Geregister_disconnect_callback
- G deregister_event_request
- Geregister_message_callback
- Gregister_transition_callback
- 🗇 disconnect
- G disconnect_from_other_client
- get_message_facilities
- get_midas_version
- get_recent_messages
- get_triggered_alarms
- hist_get_data
- hist_get_events
- hist_get_recent_data
- hist_get_tags

- 🕅 msg
- 🗘 odb_delete
- Odb_exists
- 🗘 odb_get
- Odb_get_link_destination
- Odb_last_update_time
- O odb_link
- 🛇 odb_rename
- 🗘 odb_set
- Odb_watch
- ♀ open_event_buffer
- 🗘 pause_run

- receive_event
- ☆ register_deferred_transition_callback
- ☆ register_disconnect_callback
- register_event_request
- register_jrpc_callback
- register_message_callback
- ☆ register_transition_callback
- 🗘 reset_alarm
- 🕅 resume_run
- 🛇 send_event
- Set_transition_sequence
- 🗇 start_run
- 🗇 stop_run
- 😚 trigger_internal_alarm

Example use - data analysis

- Lots of students only know python/numpy
- For several experiments I convert midas events into experiment-specific data structures

reader = pol_data.PolFile(file_path)

for pol_event in reader:
 print(pol_event.keysight_voltage)

 One also has a custom analysis GUI in python/Tk

Port Run File Here: Run Information: MCS# Info: Extract/Convert Variate Soan Type Code Sive Data Port Te ² Ref Wavenumber: Oyde Counter Sive Data Port Te ² Sive Data 13923.5 Supergode Counter Sive Data Port Te ² Sive Data 13923.5 Oyde Soc Counter Sive Data Port Te ² Sive Data Adomine Mass: Oyde Soc Counter Sive Data Port Te ² Sive Data Adomine Mass: Oyde Soc Counter Sive Data Port Te ² RFO Voltage: Port Te ² Import runxxxxx.mid file Siveped Oyde Counter Sive Data Port Te ² Sive Data 19985.0 Clear Fig Kokyeed Counter Sive Data Port Te ² Sive Data 13917.455 Clear Fig KOKyeet Sive Data Port Te ² Sive Data 5.3 ADC(1) Input Sive Data Port Te ² Sive Data 5.3 ADC(2) Input Sive Data 0 6 ADC(2) Input Sive Data Port Te ² Sive Data<	Information: MCS# Info: Extract/Convert Variables: e code Save Data PMT Ref Wavenumber: ounter Save Data Pkt ToF Save Data 13923.5 Counter Save Data Pkt ToF Save Data Atomic Mass: Counter Save Data Pkt ToF Save Data Atomic Mass: Counter Save Data Pkt ToF quency Vount Save Data 205.988670 Counter Save Data Pkt ToF quency Vounts Save Data 19985.0 Gordiner Save Data Pkt ToF Quency Vounts Save Data 19985.0 Gragmend Save Data Pkt ToF Quency Vounts Save Data 13917.455 Kolt Save Data Pkt ToF Quency Vounts Save Data 13917.455 Kolt Save Data Pkt ToF Quency Vounts Save Data 50.3 Input Save Data Pkt ToF Quency Vounts Save Data 60.0 Input Save Data Pkt ToF Quency Vounts Save Data 0.0 Input Save Data Pkt ToF Quency Vounts Save Data 0.0 Input Save Data Pkt ToF Quency Vounts Save Data 0 600 (Q) Input Save Data Pkt ToF Quency Vounts<	•			Run Viewer			
Sean Type Code Save Data PMT Ref Wavenumber: Qcbc Counter Save Data Pot ToF Save Data 13923.5 Supergole Counter Save Data Pot DAC Vot v Counts Save Data Atomic Mass: Opde/sSC Counter Save Data Pot ToF Save Data Atomic Mass: Import runxxxx.mid file Skep Counter Save Data Pot ToF Save Data 19985.0 If Cycles Halogrammed Save Data Pot ToF Save Data 19985.0 19985.0 If Cycles Halogrammed Save Data Pot ToF Save Data 13917.455 Clear Fig DACVeit Save Data Pot ToF Save Data 13917.455 ADC(1) Input Save Data Pot ToF Save Data 13917.455 ADC(2) Input Save Data Pot ToF Save Data 1000000000000000000000000000000000000	e Code Save Data PMT FF Save Data PR TF Counter Save	port Run File Here:	Run Informati	on:	MCS# Inf	o:	Extract/Convert Variables:	
Ocide Counter Save Data Piot ToF Save Data 13923.5 Supercycle Counter Save Data Piot ToF Save Data Atomic Mass: Opdes/SC Counter Save Data Piot ToC Volt Counts Save Data 2005.998670 Import runxxxxx.mid file Skepe Data Piot Frequency v Counts Save Data Piot ToF Save Data 19985.0 If Cycles Histogrammed Save Data Piot DAC Volt v Counts Save Data Laser Lock Wavenumber: Saan Increment Counter Save Data Piot DAC Volt v Counts Save Data MCP3 Kepec Ocunter ADQD Input Save Data Piot ToF Save Data MCP3 Kepec Ocunter ADQD Input Save Data Piot DAC Volt v Counts Save Data MCP3 Kepec Ocultariation: ADQD Input Save Data Piot ToF Save Data Piot ToF Save Data O ADQD Input Save Data Piot DAC Volt v Counts Save Data O O O ADQD Input Save Data Piot DAC Volt v Counts Save Data	save Data Piot ToF Save Data 13923.5 Counter Save Data Piot DAC Volt v Courts Save Data Atomic Mass: Counter Save Data Piot Frequency v Courts Save Data 205.998670 Counter Save Data Piot Frequency v Courts Save Data 205.998670 Counter Save Data Piot Frequency v Courts Save Data 19985.0 orgammed Save Data Piot DAC Volt v Courts Save Data 19985.0 orgammed Save Data Piot DAC Volt v Courts Save Data 13917.455 rtt Counter Save Data Piot Frequency v Courts Save Data 13917.455 rtt Counter Save Data Piot Frequency v Courts Save Data 13917.455 rtt Counter Save Data Piot Frequency v Courts Save Data 0.0 input Save Data Piot Frequency v Courts Save Data 0.0 input Save Data Piot Frequency v Courts Save Data 0 600 (Zi) Input Save Data Piot Frequency v Courts Piot Frequency v Courts Piot Frequency v Courts Piot Frequency v Courts (Zi) Input Save Data Piot Frequency v Courts 2.0 Piot Frequency v Courts Piot Frequ		Scan Type Code	Save Data	PMT		Ref Wavenumber:	
Supercycle Courter Save Data Pfot DAC Volt V Courts Save Data Atomic Mass: Oplest/SC Courter Save Data Pot Frequency v Courts Save Data 205:998670 Imported tun00085.mid.124 Swep Courter Save Data Other requency v Courts Save Data 205:998670 Import runxxxx.mid file Skeped Courter Save Data Pot ToF Save Data 19985.0 If Cycles Histogrammed Save Data Pot DAC Volt V Courts Save Data 19917.455 Clear Fig DAC Volt Save Data Pot Frequency v Courts Save Data 19917.455 Clear Fig DAC Volt Save Data Pot Frequency v Courts Save Data 19917.455 ADC(1) Input Save Data Pot Frequency v Courts Save Data 19917.455 ADC(2) Input Save Data Pot Frequency v Courts Save Data 19917.455 ADC(2) Input Save Data Pot Frequency v Courts Save Data 0 0 ADC(2) Input Save Data Pot Frequency v Courts Save Data 0 0	Counter Save Data Piot DAC Volu V Counts Save Data Atomic Mass: Counter Save Data Piot Frequency v Counts Save Data 205.998670 Counter Save Data Piot Frequency v Counts Save Data 205.998670 Counter Save Data Piot Frequency v Counts Save Data 10995.0 orgrammed Save Data Piot ToF Save Data 11997.455 orgrammed Save Data Piot DAC Vol v Counts Save Data 13917.455 volt Save Data Piot Frequency v Counts Save Data 13917.455 volt Save Data Piot Frequency v Counts Save Data 13917.455 volt Save Data Piot Frequency v Counts Save Data 13917.455 volt Save Data Piot Frequency v Counts Save Data 13917.455 volt Save Data Piot DAC Volt v Counts Save Data 100.0 riput Save Data Piot Frequency v Counts Save Data 0.0 600 vigit piut Save Data Piot DAC Volt v Counts Save Data 0 600 Vigit piut Save Data Piot DAC Volt v Counts 2.0 Piot Piot Piot Piot Piot Piot Piot Piot		Cycle Counter	Save Data	Plot ToF	Save Data	13923.5	
Opdaes/SC Counter Save Data Pod Frequency v Counts Save Data 205:998670 Imported L0000086/mid.I24 Sweep Counter Save Data Channel 22 CRPC Voltage: Import runxxxx.mid file Skipped Opide Counter Save Data Piot ToF Save Data Channel 22 Clear Fig C/Cpdes Heldsgrammed Save Data Piot ToF Save Data Latser Lock Wavenumber: Scan Increment Counter Save Data MCP3 Kopcoc Calibration: Clear Fig DACVolt Save Data MCP3 Kopcoc Calibration: ADCQ1 Input Save Data Piot DAC Volt Counts Save Data Hort ToF ADCQ1 Input Save Data Piot DAC Volt Counts Save Data Channel 24 ADCQ1 Input Save Data Piot DAC Volt Counts Save Data 0 0 ADCQ3 Input Save Data Piot DAC Volt Counts Save Data 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 <t< td=""><td>Counter Save Data Pot Frequency v Counts Save Data 205.998670 Save Data Pot ToF Save Data 19985.0 le Counter Save Data Pot ToF Save Data 19985.0 ogrammed Save Data Pot ToF Save Data 19985.0 ogrammed Save Data Pot DAC Vok v Counts Save Data 19985.0 rot Counter Save Data Pot DAC Vok v Counts Save Data 13917.455 variation Save Data Pot ToF roguency v Counts Save Data 13917.455 variation Save Data Pot ToF Save Data 13917.455 input Save Data Pot ToF Save Data 100.0 input Save Data Pot ToF variations Save Data 0.0 input Save Data Pot ToF variations Save Data 0.0 input Save Data Pot ToF variations Save Data 0.0 input Save Data Pot ToF variations Save Data 0.0 (20) input Save Data Pot ToF variations Save Data 0 600 (20) input Save Data Pot ToF variations Save Data 0 600 (20) input Save Data</td><td></td><td>Supercycle Counter</td><td>Save Data</td><td>Plot DAC Volt v Counts</td><td>Save Data</td><td>Atomic Mass:</td></t<>	Counter Save Data Pot Frequency v Counts Save Data 205.998670 Save Data Pot ToF Save Data 19985.0 le Counter Save Data Pot ToF Save Data 19985.0 ogrammed Save Data Pot ToF Save Data 19985.0 ogrammed Save Data Pot DAC Vok v Counts Save Data 19985.0 rot Counter Save Data Pot DAC Vok v Counts Save Data 13917.455 variation Save Data Pot ToF roguency v Counts Save Data 13917.455 variation Save Data Pot ToF Save Data 13917.455 input Save Data Pot ToF Save Data 100.0 input Save Data Pot ToF variations Save Data 0.0 input Save Data Pot ToF variations Save Data 0.0 input Save Data Pot ToF variations Save Data 0.0 input Save Data Pot ToF variations Save Data 0.0 (20) input Save Data Pot ToF variations Save Data 0 600 (20) input Save Data Pot ToF variations Save Data 0 600 (20) input Save Data		Supercycle Counter	Save Data	Plot DAC Volt v Counts	Save Data	Atomic Mass:	
Import sd un00286.mid.iz4 Sweep Counter Save Data Channel 22 RFC Vottage: Import runxcocx.mid file Skeped cycle Counter Save Data Piot ToF Save Data 19985.0 # Cycles Histogrammed Save Data Piot ToF Save Data Laser Lock Wavenumber Scan increment Counter Save Data Piot ToF (Save Data Laser Lock Wavenumber Clear Fig NoCvet Save Data Piot Frequency voorts Save Data 13917.455 ADC(0) Input Save Data Piot Frequency voorts Save Data Harrison Voltage: ADC(2) Input Save Data Piot ToF Save Data Harrison Voltage: ADC(2) Input Save Data Piot ToF Save Data O.0 ADC(3) Input Save Data Piot ToF Save Data 0.0 AVeraged ADC(2) Input Save Data Piot ToF Voltage: 0 Figure 1 Averaged ADC(2) Input Save Data Piot ToG voltage 0 Figure 1 Averaged ADC(2) Input Save Data Piot ToG voltage Figure 1 Figure 1 Averaged ADC(2) Input Save Data Piot ToG voltage 2.0 Figure 1 Averaged ADC(2) Input Save Data Piot ToG voltage 1.0 1.0 1.0 </td <td>Save Data Channel 22 RFQ Voltage: lc Cunter Save Data PRt FG Save Data 19995.0 orgarmed Save Data PRt FAC Voltage: 19995.0 orgarmed Save Data PRt FAC Voltage: 13917.455 save Data PRt FACutor Counts Save Data 13917.455 save Data PRt FRequency Vounts Save Data 13917.455 input Save Data PRt FRequency Vounts Save Data 60.3 input Save Data PRt Frequency Vounts Save Data 14171500 Voltage: input Save Data PRt Frequency Vounts Save Data 0.0 input Save Data PRt Frequency Vounts Save Data 0.0 input Save Data PRt Frequency Vounts Save Data 0.0 input Save Data PRt Frequency Vounts Save Data 0 600 XC[1] input Save Data PRt Frequency Vounts Prepare 1 Prepare 1 XC[2] input Save Data PRt Frequency Vounts 2.0 Figure 1 XC[2] input Save Data PRt Frequency Vounts 1.0 Figure 1 XC[2] input Save Data PRt Frequency Vounts 1.0 1.0 I.0</td> <td></td> <td>Cycles/SC Counter</td> <td>Save Data</td> <td>Plot Frequency v Counts</td> <td>Save Data</td> <td>205.998670</td>	Save Data Channel 22 RFQ Voltage: lc Cunter Save Data PRt FG Save Data 19995.0 orgarmed Save Data PRt FAC Voltage: 19995.0 orgarmed Save Data PRt FAC Voltage: 13917.455 save Data PRt FACutor Counts Save Data 13917.455 save Data PRt FRequency Vounts Save Data 13917.455 input Save Data PRt FRequency Vounts Save Data 60.3 input Save Data PRt Frequency Vounts Save Data 14171500 Voltage: input Save Data PRt Frequency Vounts Save Data 0.0 input Save Data PRt Frequency Vounts Save Data 0.0 input Save Data PRt Frequency Vounts Save Data 0.0 input Save Data PRt Frequency Vounts Save Data 0 600 XC[1] input Save Data PRt Frequency Vounts Prepare 1 Prepare 1 XC[2] input Save Data PRt Frequency Vounts 2.0 Figure 1 XC[2] input Save Data PRt Frequency Vounts 1.0 Figure 1 XC[2] input Save Data PRt Frequency Vounts 1.0 1.0 I.0		Cycles/SC Counter	Save Data	Plot Frequency v Counts	Save Data	205.998670	
Import runxxxx.mid file Skipped Qyde Counter Save Data Pixt ToF Save Data 19985.0 I* Cycles Histogrammed Save Data Pixt ToF Vok V Courts Save Data Lasser Lock Wayenumber Scan Increment Counter Save Data Pixt ToF Vok V Courts Save Data Lasser Lock Wayenumber Clear Fig DACVAI Save Data Pixt ToF Save Data MCP3 ADCQ1 piput Save Data Pixt ToF Save Data MCP3 Kopco Collination ADCQ1 piput Save Data Pixt ToF Save Data MCP3 Clear Fig ADCQ1 piput Save Data Pixt ToF Save Data Harrison Voltage: ADCQ1 piput Save Data Pixt ToF Save Data O.0 ADCQ1 piput Save Data Pixt ToF Save Data O.0 AVeraged ADCQ1 piput Save Data Pixt ToF Save Data O Averaged ADCQ1 piput Save Data Pixt ToF Save Data O Figuret TRULS177/METER-LAMBDA1 Save Data Truls 177/METER-LAMBDA1	Save Data Pict ToF Save Data 19985.0 orgarmed Save Data Pict DAC Vol v Courts Save Data Laser Lock Wavenumber: int Counter Save Data Pict TAC Vol v Courts Save Data 13917.455 int Counter Save Data Pict TAC Vol v Courts Save Data 13917.455 int Counter Save Data Pict TAC Save Data 13917.455 introd Save Data Pict TAC Save Data 60.3 input Save Data Pict TAC Vol v Courts Save Data 14arrison Voltage: input Save Data Pict Trequency v Counts Save Data 0.0 input Save Data Pict TAC Vol v Counts Save Data 0.0 input Save Data Pict TAC Vol v Counts Save Data 0.0 input Save Data Pict TAC Vol v Counts Pice TAC Vol v Counts Pice TAC Vol v Counts Cipi Input Save Data Pict TAC Vol v Counts Pice TAC Vol v Counts Pice TAC Vol v Counts Cipi Input Save Data Pict Frequen	Imported run00086.mid.lz4	Sweep Counter	Save Data	Channel 22		RFQ Voltage:	
# Cycles Histogrammed Save Data Piot DAC Volt V Courts Save Data 13917.455 Ciear Fig DAC Volt Save Data MCP3 Kapeo Calibration: ACQ0I Input Save Data Piot Trequency v Courts Save Data MCP3 ACQ0I Input Save Data Piot ToF Save Data Harrison Voltage: ADQ13 Input Save Data Piot ToF Save Data 0.0 ADQ23 Input Save Data Piot ToF Save Data 0.0 ADQ23 Input Save Data Piot ToF Save Data 0.0 0 AVeraged ADQ11 Input Save Data Piot DAC Volu V Courts Save Data 0	orgammed Save Data Piot DAC Volt v Counts Save Data Laser Lock Wavenumber: ntr Counter Save Data Piot Frequency v Counts Save Data 13917.455 Non Save Data MCPO Kepco Calibration: Input Save Data Piot Frequency v Counts Save Data 0.3 Input Save Data Piot ToF Save Data 400.0 Input Save Data Piot ToF Save Data 0.0 Input Save Data Piot ToF Save Data 0.0 Save Data Piot Frequency v Counts Save Data 0 600 Ci[0] Input Save Data Piot Frequency v Counts Piot Frequency v Counts Piot Frequency v Counts Ci[0] Input Save Data Piot Frequency v Counts Piot Frequency v Counts Piot Frequency v Counts Ci[0] Input Save Data Piot Frequency v Counts Piot Frequency v Counts Piot Frequency v Counts Ci[0] Input Save Data Piot Frequency v Counts Piot Frequency v Counts Piot Frequency v Counts Ci[0] Input Save Data Piot Frequency v Counts Piot Frequency v Counts Piot Frequency v Counts Save Data Save Data O O O Ci[0] Input <td< td=""><td>Import runxxxxx.mid file</td><td>Skipped Cycle Counter</td><td>Save Data</td><td>Plot ToF</td><td>Save Data</td><td>19985.0</td></td<>	Import runxxxxx.mid file	Skipped Cycle Counter	Save Data	Plot ToF	Save Data	19985.0	
Stan Increment Counter Save Data Pot Frequency v Counts Save Data MCP3 Kepco Calibration: AAC Vait Save Data Pot Frequency v Counts Save Data Pot OF Save Data Sove Data <	ext Counter Save Data Piot Frequency v Counts Save Data 13917.455 Volt Save Data MCP3 Kepco Calibration: Input Save Data Piot ToF Save Data 60.3 Input Save Data Piot DAC Volt v Counts Save Data Harrison Voltage: Input Save Data Piot DAC Volt v Counts Save Data 0.0 Input Save Data Piot ToF Save Data 0.0 Kittle Save Data Piot ToF Save Data 0 Kittle Save Data Piot ToF Save Data 0 Kittle Save Data 1.5 1.0 1.0 FILAMBDA Save Data 0.5 1.0 1.0 1.0		# Cycles Histogrammed	Save Data	Plot DAC Volt v Counts	Save Data	Laser Lock Wavenumber:	
Clear Fig DAC Volt Save Data MCP3 Kepco Callbration: AOQQI Input Save Data Pict ToF Save Data Fict ToF Save Data Harrison Voltage: AOQQI Input Save Data Pict ToF Volt Volt Volt Courts Save Data Harrison Voltage: AOQQI Input Save Data Pict ToF Volt Volt Volt Courts Save Data Harrison Voltage: AOQQI Input Save Data Pict Frequency v Courts Save Data 0.0 Averaged ADQQI Input Save Data Pict ToF Save Data 0 6 Averaged ADQQI Input Save Data Pict ToF Save Data 0 6 Averaged ADQQI Input Save Data Pict ToF Save Data 0 6 Averaged ADQQI Input Save Data Pict ToF Save Data 0 6 Averaged ADQII Input Save Data Pict Frequency v Courts 2.0 7 Averaged ADQII Input Save Data Pict Frequency v Courts 2.0 1.5 1.6 1.6 1.6 1.6 1.	Noti Save Data MCPO Kepco Calibration: Input Save Data Port DF Save Data 50.3 Input Save Data Port DAC Volv Courts Save Data Harrison Voltage: Input Save Data Port Frequency v Courts Save Data 0.0 Input Save Data Port Frequency v Courts Save Data 0.0 KCI0 Input Save Data Port Frequency v Courts Save Data 0 600 XCI0 Input Save Data Port Frequency v Courts Person Person XCI0 Input Save Data Port Frequency v Courts Person Person XCI0 Input Save Data Port Frequency v Courts Person Person XCI0 Input Save Data Port Frequency v Courts Person Person XCI0 Input Save Data Port Frequency v Courts Person Person XCI0 Input Save Data Port Frequency v Courts Person Person XCI0 Input Save Data Port Frequency v Courts Person Person XCI0 Input Save Data Port Frequency v Courts Person Person XCI0 Input Save Data Port Frequency v Courts Person Person		Scan Increment Counter	Save Data	Plot Frequency v Counts	Save Data	13917.455	
ADCQ0 Input Save Data Piot ToF Save Data 50.3 ADCQ1 Input Save Data Piot DAC Volt V Courts Save Data Harrison Voltage: ADCQ2 Input Save Data Piot AC Volt V Courts Save Data 0.0 ADCQ3 Input Save Data Piot Frequency v Courts Save Data 0.0 ADCQ3 Input Save Data Piot Frequency v Courts Save Data 0.0 Averaged ADCQ3 Input Save Data Piot DAC Volt V Courts Foural Averaged ADCQ3 Input Save Data Piot AC Volt V Courts Foural Averaged ADCQ3 Input Save Data Piot AC Volt V Courts Foural Averaged ADCQ3 Input Save Data Piot Frequency v Courts Foural Averaged ADCQ3 Input Save Data Piot Frequency v Courts Foural Averaged ADCQ3 Input Save Data Piot Frequency v Courts Foural TRILIST77METER:LAMBDA3 Save Data 1.0 Intel Field AD Volt V Courts Foural TRILIST77METER:LAMBDA4 Save Data 1.0 Intel Field AD Volt V Courts Intel Field AD Volt V Courts Mevigint Voltage Save Data 0.5 Intel Field AD Volt V Courts Intel Field AD Volt V Courts	Input Save Data Piot ToF Save Data 50.3 Input Save Data Piot DAC Volv Courts Save Data Harrison Voltage: Input Save Data Piot Frequency v Courts Save Data 0.0 Input Save Data Piot Frequency v Courts Save Data 0.0 (C)I prut Save Data Piot DAC Volv v Courts Save Data 0 600 C(2)I prut Save Data Piot DAC Volv v Courts Imput Figure 1 Figure 1 C(2)I prut Save Data Piot DAC Volv v Courts Imput Figure 1 Figure 1 C(2)I prut Save Data Piot DAC Volv v Courts Imput Figure 1 Figure 1 C(2)I prut Save Data Piot Frequency v Courts Imput Figure 1 Figure 1 C(2)I prut Save Data Piot Frequency v Courts Imput Figure 1 Figure 1 C(2)I prut Save Data Piot Frequency v Courts Imput Imput Imput C(2)I prut Save Data Piot Frequency v Courts Imput Imput Imput Save Data Piot Frequency Imput Imput Imput Imput Piot Piot Frequency Imput Imput Imput	Clear Fig	DAC Volt	Save Data	MCP3		Kepco Calibration:	
ADC[1] Input Save Data Pict DAC Volt v Courts Save Data Harrison Voltage: ADC[2] Input Save Data Pict Frequency v Courts Save Data 0.0 ADC[3] Input Save Data Channel 24 ToF Window Cut (bins): Averaged ADC[0] Input Save Data Pict ToF Save Data 0 6 Averaged ADC[2] Input Save Data Pict ToF Save Data 7 Figure 1 7 Averaged ADC[2] Input Save Data Pict DAC Volt Courts ####################################	Input Save Data Piot DAC Vol v Courts Save Data Hatrison Voltage: Input Save Data Piot Frequency v Courts Save Data 0.0 Input Save Data Channel 24 ToF Window Cut (bins): CQD Input Save Data Piot ToF Save Data 0 CQD Input Save Data Piot ToF Save Data 0 CQD Input Save Data Piot ToF Save Data 0 CQD Input Save Data Piot ToF Save Data Piot ToF CQD Input Save Data Piot ToF Save Data Piot ToF CQD Input Save Data Piot ToF Save Data Piot ToF CQD Input Save Data Piot Frequency v Counts 2.0 Input ERLAMBDA1 Save Data 1.5 Input Input Save Data Save Data 0.5 Input Input Value Save Data 0.5 Input Input		ADC[0] Input	Save Data	Plot ToF	Save Data	50.3	
ADC(2) Input Save Data Ped Frequency v Counts Save Data 0.0 ADC(3) Input Save Data Ped Frequency v Counts 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Input Save Data Pot Frequency v Courts Save Data 0.0 Input Save Data Channel 24 ToF Window Cut (bins): XC[0] Input Save Data Pot ToF Save Data 0 600 XC[0] Input Save Data Pot ToF Save Data 0 600 XC[0] Input Save Data Pot Frequency v Courts Pages 1 Pages 1 XC[0] Input Save Data Pot Frequency v Courts Pages 1 Pages 1 XC[0] Input Save Data Pot Frequency v Courts 2.0 Pages 1 Pages 1 XC[0] Input Save Data Interview 1.0 1.5 1.0 <t< td=""><td rowspan="3"></td><td>ADC[1] Input</td><td>Save Data</td><td>Plot DAC Volt v Counts</td><td>Save Data</td><td>Harrison Voltage:</td></t<>		ADC[1] Input	Save Data	Plot DAC Volt v Counts	Save Data	Harrison Voltage:	
ADC[3] Input Save Data Channel 24 ToF Window Cut (bins): Averaged ADC[0] Input Save Data Piot ToF Save Data 0 6 Averaged ADC[1] Input Save Data Piot DAC Vot v Courts Piot Piot Piot Piot DAC Vot v Courts Piot Piot Piot DAC Vot v Courts Piot Piot Piot DAC Vot v Courts Piot Piot Piot Piot Piot Piot Piot Piot	Input Save Data Channel 24 ToF Window Cut (bins): C(0) Input Save Data Piot ToF Save Data 0 600 C(1) Input Save Data Piot DaC Vol v Counts Pipue 3 Pipue 3 C(2) Input Save Data Piot ToF Save Data 0 600 C(2) Input Save Data Piot ToF Save Data Pipue 3 Pipue 3 C(2) Input Save Data Piot Frequency v Counts Pipue 3 Pipue 3 Pipue 3 C(2) Input Save Data Piot Frequency v Counts 2.0 Input 3 Input 4 Input		ADC[2] Input	Save Data	Plot Frequency v Counts	Save Data	0.0	
Averaged ADC(0) Irput Save Data Pict DF Save Data 0 6 Averaged ADC(1) Irput Save Data Pict DAC Volt V Courts Figure 1 Figure 1 Averaged ADC(3) Irput Save Data Pict Frequency V Courts Figure 1 Figure 1 Averaged ADC(3) Irput Save Data Pict Frequency V Courts Figure 1 Figure 1 TRILIST/TMETERLAMBDA Save Data Save Data 1.5 1 Figure 1 TRILIST/TMETERLAMBDA Save Data 1.5 1 Figure 1 Figure 1 TRILIST/TMETERLAMBDA Save Data 1.0 1 Figure 1 Figure 1 ILE2LAS/RDPOWER Save Data 0.5 Figure 1 Figure 1 Figure 1 Wew Variations in Readback Vidages 0.5 Figure 1 Figure 1 Figure 1	C(20) Input Save Data Pict ToF Save Data 0 600 C(21) Input Save Data Pict ToF Save Data Pict ToF C(21) Input Save Data Pict Frequency v Counts Pict ToF Pict ToF C(23) Input Save Data Pict Frequency v Counts Pict ToF Pict ToF ER1.AMBDA1 Save Data 2.0 Interview Pict ToF ER1.AMBDA3 Save Data 1.5 Interview Pict ToF PPOWER Save Data 0.5 Interview Pict ToF Save Data 0.5 Interview Pict ToF Pict ToF		ADC[3] Input	Save Data	Channel 24		ToF Window Cut (bins):	
Averaged ADC[1] Irput Save Data Pot DAC Vol V Counts Figure 1 Averaged ADC[2] Irput Save Data Pot Frequency v Counts 2.0 Averaged ADC[3] Irput Save Data 1.5 1.5 TRILS177/METERLAMBDA Save Data 1.5 1.6 TRILS177/METERLAMBDA Save Data 1.0 1.0 LE2LAS/RDPOWER Save Data 0.5 1.6 Keysight Voltages Save Data 0.5 1.6	XC[1] Input Save Data Pot DAC Vola V Counts XC[2] Input Save Data Pot Frequency v Counts ZC[3] Input Save Data 2.0 FR1.AMBDA2 Save Data 1.5 ER1.AMBDA4 Save Data 1.5 Save Data 1.0 1.0 PrOVEFN Save Data SWD Save Data SWA 0.5		Averaged ADC[0] Input	Save Data	Plot ToF	Save Data	0 600	
Averaged ADC(2) Irput Save Data Pot Frequency v Counts Averaged ADC(3) Irput Save Data 2.0 TRILIST/TMETER-LAMBDA4 Save Data 1.0 TRILIST/TMETER-LAMBDA4 Save Data 1.0 ILE21AS:ROPOWER Save Data 0.5 Keysight Voltage Save Data View Variationes 0.5	CQ2] Input Save Data Pot Frequency v Counts CQ3] Input Save Data 2.0 ER1AMBDA1 Save Data 1.5 ER1AMBDA4 Save Data 1.0 PrOVEFR Save Data 0.5 Save Data 0.5		Averaged ADC[1] Input	Save Data	Plot DAC Volt v Counts		Figure 1	
Averaged ADC(3) input Save Data 2.0 TRILIS177/METERLAMBDA1 Save Data 1.5 TRILIS177/METERLAMBDA2 Save Data 1.5 TRILIS177/METERLAMBDA3 Save Data 1.0 ILE2LAS/ROPOWER Save Data 0.5 NEUTASYM Save Data 0.5 View Variations Save Data 0.5	CQ3 Input Save Data 2.0 ERLAMEDA1 Save Data 1.5 ERLAMEDA2 Save Data 1.0 FRIAMEDA4 Save Data 1.0 Srow Data 0.5 1.0		Averaged ADC[2] Input	Save Data	Plot Frequency v Counts	C		
TRILIS177/METERI.LAMBDA1 Save Data TRILIS177/METERI.LAMBDA2 Save Data TRILIS177/METERI.LAMBDA3 Save Data TRILIS177/METERI.LAMBDA4 Save Data TRILIS177/METERI.LAMBDA4 Save Data ILIEL157/METERI.LAMBDA4 Save Data NEUTASYM Save Data NEUTASYM Save Data Vew Variators Save Data Vew Variators Save Data Vew Variators Save Data	ERLAMEDA1 Save Data ERLAMEDA2 Save Data ERLAMEDA3 Save Data Save Data SYM Save Data Office Save Data SYM Save Data Office Save Data		Averaged ADC[3] Input	Save Data		2.0		
TRILIS177/METERIAMBDA2 Save Data 1.5 - TRILIS177/METERIAMBDA3 Save Data 1.0 - TRILIS177/METERIAMBDA4 Save Data 1.0 - ILE2LAS.RIPOWER Save Data 0.5 - NeUTASYM Save Data 0.5 - Keysight Voltage Save Data 0.5 -	ERLAMBDA2 Save Data 1.5 - ERLAMBDA4 Save Data 1.0 - SYM Save Data 0.5 - Sym Data 0.5 - Control of the save Data 0.5 - Contro		TRILIS177:METER:LAMBDA1	Save Data				
TRILISI77/METERLAMBDA3 Save Data 1.0 ILE2LAS:PDPOWER Save Data 1.0 NEUTASYM Save Data 0.5 Kaysight Voltage Save Data 0.5 Vew Variations in Readback Voltages 1	ERLAMEDA3 Save Data ERLAMEDA4 Save Data PrOWER Save Data SYM Save Data 0.5 -		TRILIS177:METER:LAMBDA2	Save Data		1.5	***********************	
TRILISI77/METERLAMBONA Save Data 1.0 LE2LAS:RDPOWER Save Data 0.5 Kaysigit Voltage Save Data 0.5 Vew Variations in Readback Voltages 1.0	ERLAMBDA4 Save Data DPOWER Save Data SYM Save Data Viace Save Data		TRILIS177:METER:LAMBDA3	Save Data				
ILE2LAS:RDPOWER Save Data NEUTASYM Save Data Keysight Voltage Save Data Vew Variations in Readback Voltages	Save Data Save Data SYM Save Data whate 0.5		TRILIS177:METER:LAMBDA4	Save Data		1.0		
NEUT/ASYM Save Data Keysight Voltage Save Data View Variations in Readback Voltages	SYM Save Data 0.5 -		ILE2:LAS:RDPOWER	Save Data				
Keysight Voltage Save Data View Variations in Readback Voltages	Voltage Save Data		NEUT:ASYM	Save Data		0.5	┋╘╘╘╘╘╘╘╘╘╘╘╘╘╘╘╘╘╘╘	
View Variations in Readback Voltages	*******		Keysight Voltage	Save Data		0.5		
	ions in Readback Voltages		View Variations in Readback	Voltages		0.0	*********************	
						0.0 {		

Example use - frontend

Many slow control devices use LXI/vxi-11 protocol

- E.g. Tektronix/LeCroy scopes/function generators
- Python has a vxi11 package available on PyPi
 - pip install python-vxi11, and you don't have to worry about the low-level details
- Frontend in python doesn't require much code to be written (and performance isn't important in this case)
- Could all be done in C++, but quicker for me to implement in python

Example use - complex configuration

- Many of my experiments do complex computations before configuring a device
 - E.g. computing RF frequencies and modulation based on human-understandable inputs
- Students often send me the calibration routines in python
 - May as well keep that code rather than re-writing!
- I write custom webpages that show the input and output
 - Use RPC and/or ODB hotlinks
 - Much nicer to report problems early, rather than waiting until user tries to start a run

Example use - PPG compiler

- PPG is a 32-channel timing sequencer used at TRIUMF. Accepts low-level bytecode.
- Users wanted a simple way to program it
- Python frontend/compiler and webpage!
 - Pulses/loops etc stored in ODB
 - Webpage issues RPC to get a visual display of the sequence
 - Bytecode generated at run start
- Compiler would have been much more tedious to write in C++

external tri	ger 🔽 Use external d	ock Define channel names Scan pa	irameters Save and load parameters			
may use for erscores, an	mulae to specify PPG bloc d must not start with a nu	k time offsets and pulse widths. Variable mber.	s used in the formula can be defined here. Va	riable names can only contain letters, nur		
 Allowed i Allowed i Example Pre-defin pi c = tou Custom v 	sperators: + - * / ^ () math functions: sin, co formula (if you define a v ed variables: 3.14192653380793 2.718281826499045 = 6.283185307179586 ariables: (Add new variab	is, tan, asin, acos, atan, abs, exp annable x): (sin(x)>2 + cos(x)∧2) / m	s, pow, sqrt, log, log2, log10 2.5			
	Block name	Timing	Action	Tools		
0	trans_PLT_up	0.5 ms after End of previous block v	toggle [PLT (CH12) *	(Benarie) (Delete) Move: Add above: Add below: Mute: (genre)		
1	begin_ramp	0.001 ms after End of previous block *	start looping 3 times	(Barares) Dolota Move (Add above) (Add below) (Muter (grows)		
2	pulse_EBITtrip	0.001 ms after [Start of each loop of ramp	pulse Trigger EBIT (CH3) v for 0.01 ms	(Revare) Doles (Meve) Add above) Add beine) MAR (spraw)		
3	pulse_MRToFtrig	0.001 ms after [Start of each loop of ramp v]	pulse Trigger MRTOF (CH2)	(Annarie) (Dolota) (Move) (Add above) (Add below) (M.K.) (spore)		
4	pulse_trapv_injection	0.001 ms after [End of pulse_mntoftrig v]	pulse Trap voltage step (CH1) $ \times $ for 0.05 ms	(Amarie) (Delete) (Move) (Add above) (Add below) (Mitte) (spore)		
5	pulse_freq_step	0.005 ms after Start of each loop of ramp v	pulse: Quad freq step (CH10)	(Benarie) (Dolote) (Move) (Add above) (Add below) (MAK) (sprove)		
6	pulse_MBR_down	2.5 ms after Start of each loop of ramp v	pulse MBR (CH11) v for 0.005 ms	(Resarce) (Dollar) (Merce) (Add above) (Add below) (Mate) (specer)		
7	pulse_ISAC_gate	1 ms after Start of each loop of ramp v	pulse ISAC gate (CH8) v for 12 ms	Recard Dates (Move Add above Add below Mars (grave		
8	pulse_TSYBL	0.01 ms after Start of pubse_mbr_clown v	pulse TSYBL (CH4) v for 0.01 ms	Reserve) Delette Move: Add above: Add below: Mass (grove)		
9	pulse_TOPGate	0.066 ms after Start of pulse_mbr_down v	pulse TOF gate (CH9) v for 0.0015 ms	Benarme Delette Move Add above Add below Mate (growe		
10	multi_caen	0.5 ms after Start of pulse_tofgate v	pulse CAEN trigger (CH15) v for 0.001 ms 8 times, with 0.05ms delay between start of each pulse	(Annure) (Delete) Move: Add above: Add below) (MAR) (grove)		
11	pulse_PLT_down	0.089000000000001 ms after Start of pulse,mbr,down v	pulse (PLT (CH12) v for 0.2 ms	(Annare) (Doleta) (Merce) (Add Science) (MAR) (sprace)		
12	pulse_trapv_capture	0.0975 ms after Start of pulse_mbr_clown v	pulse Trap voltage step (CH1) × for 0.05 ms	Record Dates New Addates Addates Unrus (get		
13	pulse_OPL1	110 ms after End of previous block v	pulse Dipole RF (CHB) v for 50 ms	Reverse Delete Move Add above Add below Mate (grow		
14	pulse_QUAD1	0.105000000000001 ms after End of pulse_dpt1 v	pulse Quad RF (CH7) v for 0.03 ms	(Broard) Dolota (Mrow) Add abrow (Add below) Mate (grows)		
15	pulse_trapv_ejection	100 ms after (End of previous block v)	pulse Trap voltage step (CH1) $ \times $ for 0.05 ms	(Annare) Deleth Meyes (Add above) (Add beines (MAR) (genre)		
16	pulse_TDCGate	0.002 ms after Start of first rep of multi_caen ×	pulse TDC gate (CH13) v for 0.2 ms	Reverse Delete Move Add above Add below Mate genre		



<technical details>

def odb set(self, path, contents, create if needed=True,



remove_unspecified_keys=True, resize_arrays=True, lengthen_strings=True, explicit_new_midas_type=None, update_structure_only=False):

Set the value of an ODB key, or an entire directory. You may pass in normal python values, lists and dicts and they will be converted to appropriate midas ODB key types (e.g. int becomes midas.TID_INT, bool becomes midas.TID BOOL).

Sensible defaults have been chosen for converting python types to the C types used internally in the midas ODB. However if you want more control over the ODB type, you may use the types defined in the ctypes library. For example, regular python integers become a midas.TID_INT, but you can use a `ctypes.c_uint32` to get a midas.TID_DWORD.

If you are setting the content of a directory and care about the order in which the entries appear in that directory, `contents` should be a `collections.OrderedDict` rather than a basic python dict. See the note in `odb_get` for more about dictionary ordering.

Args:

- * path (str) The ODB entry to set. You may specify a single array index if desired (e.g. "/Path/To/My/Array[1]").
- * contents (int/float/string/tuple/list/dict etc) The new value to set * create_if_needed (bool) - Automatically create the ODB entry (and parent directories) if needed.
- * remove_unspecified_keys (bool) If `path` points to a directory and `contents` is a dict, remove any ODB keys within `path` that aren't present in `contents`. This means that the ODB will exactly match the dict you passed in. You may want to set this to False if you want to only update a few entries within a directory, and want to do so with only a single call to `odb set()`.
- * resize_arrays (bool) Automatically resize any ODB arrays to match the length of lists present in `contents`. Arrays will be both lengthened and shortened as needed.
- * lengthen_strings (bool) Automatically increase the storage size of a TID_STRING entry in the ODB if it is not long enough to store the value specified. We will include enough space for a final null byte.
- * explicit_new_midas_type (one of midas.TID_xxx) If you're setting the value of a single ODB entry, you can explicitly specify the type to use when creating the ODB entry (if needed).
- * update_structure_only (bool) If you want to add/remove entries in an ODB directory, but not change the value of any entries that already exist. Only makes sense if contents is a dict / `collections.OrderedDict`. Think of it like db_check_record from the C library.

Raises:

- * KeyError if `create_if_needed` is False and the ODB entry does not already exist.
- * TypeError if there is a problem converting `contents` to the C type we must pass to the midas library (e.g. the ODB entry is a TID_STRING but you passed in a float).
- * ValueError if there is a non-type-related problem with `contents` (e.g. if `resize arrays` is False and you provided a list that doesn't match the size of the existing ODB array).
- * midas.MidasError if there is some other midas issue.

Advanced usage - API

- odb_set has parameters that let you do powerful things, especially when passing a dict
- Defaults are sensible (principle of least surprise), but options are there if you need them
- All the options are clearly documented in client.py
- Same idea for many of the functions

Advanced usage - ctypes

- ODB gives fine control over how data is stored
 - uint8_t, int16_t, float, double etc
- Python just has integer and float (really a double)
- If you care, can use ctypes library to specify exact data type you want (e.g. ctypes.c_uint32) and/or specify the midas data type (e.g. midas.TID_UINT32)

Implementation of midas.client - C side

- Python ctypes library comes as standard and can call C functions
- Midas is now compiled as C++, which is much harder to call from python
- I added a midas_c_compat.cxx/h file that provides a Ccompatible wrapper of the functions I need
 - Generally trivial, some shenanigans for functions that populate std::vector<std::string> etc (char***)
 - All exposed functions use extern "C"

Implementation of midas.client - py side

- midas.MidasLib uses ctypes' ability to call C functions
- Automatically discovers all the functions in libmidas-ccompat.so
- Intercepts all the return values from C functions
 - If not 1 (SUCCESS), raises an Exception
 - Whitelist of functions that return other codes (e.g. al_reset_alarm can return "AL_RESET") or that don't return status codes at all

Process to expose a new function

 Write a wrapper in midas_c_compat.cxx/h (with conversion between C-compatible and C++ if needed)

```
INT c_al_reset_alarm(const char *alarm_name) {
    return al_reset_alarm(alarm_name);
}
```

 Write python function in midas.client (with conversion between python and ctypes if needed)

```
def reset_alarm(self, alarm_name):
    c_name = ctypes.create_string_buffer(bytes(alarm_name, "utf-8"))
    self.lib.c_al_reset_alarm(c_name)
```

Edit MidasLib if this C function is "special" and doesn't return a status code </technical details>

Future plans

- I recently added support for accessing the history system
 - Anything you did with mhist can be done in python
- I don't think there are any major midas features left that can't be accessed through python
- Keep emailing me with requests though!
 - There are always use cases that I haven't thought of (e.g. 40kHz of events through a python frontend, getting recent messages from the log, ...)
 - I'll try to implement/improve ASAP

Summary

- Lots of progress on using python in midas since the last workshop
- Should be easy to maintain as midas evolves
- You should be able to write scripts, frontends and data analysis code all in python
- I really like having custom webpages talking to python code via JRPC
- Let me know if there are any features or improvements you'd like to see!