universal$_usbtmcDocumentation$

Release v0.3.dev

Philipp Klaus

Aug 27, 2017

Contents

Thanks for reading the documentation of the Python package universal_usbtmc.

universal_usbtmc is free software, published on Github by Philipp Klaus.

Contents:

CHAPTER 1

The base class universal_usbtmc.Instrument

The universal_usbtmc backends all derive from *universal_usbtmc.Instrument* and thus inherit or reimplement its methods.

Thus, you can read the reference for this class to find out how to use the backends:

class universal_usbtmc.**Instrument**(*device*)
USBTMC instrument interface

The backends get initialized with a single device string to set up the connection.

Parameters **device** (*str*) – The device to connect to

ENCODING = 'utf-8'
The encoding used when interpreting bytes as strings

LINE_ENDING = ''
The line ending to add to write() or query() commands

write_raw(*data*)
Send binary data to the instrument

The backends need to implement this method!

Parameters **data** (*bytes*) – The data to send to the instrument

read_raw(*num=-1*, *timeout=0.0*)
Read binary data from the instrument

The backends need to implement this method!

Parameters

- **num** (*bytes*) – Number of bytes to read back
- **timeout** (*float*) – Seconds until the read operation should time out

write(*message*, *encoding='default'*, *line_ending='default'*)
Send a string message to the instrument

Parameters

- **message** (*str*) – The message to send
- **encoding** (*str*) – The encoding to use when converting the message from str to bytes
- **line_ending** (*str*) – The line ending to add to the message

read(*num=-1*, *encoding='default'*, *line_ending='default'*)
Read a response string from the instrument

Parameters

- **num** (*bytes*) – Number of bytes to read back
- **encoding** (*str*) – The encoding to use when converting the response from bytes to str
- **line_ending** (*str*) – The line ending to strip off from end of received message

query_raw(*message*, *encoding='default'*, *num=-1*)
Convenience method to first send a **string** message and then return the **binary** response data from a subsequent `read_raw()` call.

query(*message*, *num=-1*, *encoding='default'*, *line_ending='default'*)
Convenience method to first send a string command, then read and return a response string. :param str line_ending: The line ending to add to the message

idn
The response to an `*IDN?` query.

Python Module Index

u

Index