

Python Mates Controller Library

Introduction

This library is developed to easily control Breadboard Mates modules using any device that can run Python by utilizing the [Mates Controller Command Protocol](#). This applies to projects developed using Commander and Architect environments.

Supported Devices

This library is developed for Python3 and designed to be used with any operating system as long as it is supported by the `pyserial` library.

Installation

This library can be installed from the Python Packaging Index (PyPI) by running the command:

```
pip3 install mates-controller
```

Constructors

This section serves to provide brief discussion about the constructors that can be used to initialize the library.

MatesController(portName, resetFunction, debugStream, debugFileLength)

Constructs all the necessary attributes associated with an instance of a Mates Controller Object.

Parameters	Type	Description
portName	str	the name of the port to be opened. Example: <code>/dev/ttyUSB0</code> for linux
resetFunction	function	function used to perform a hard reset
debugStream (optional)	io.TextIOWrapper	text file object to write debugging code to, supply of none will result in no debugging. Ex. <code>sys.stdout</code> , <code>open('log.txt', 'r+')</code>
debugFileLength (optional)	int	determines the extent of debug history kept with respect to lines in a file, given a circular log. 0 indicates full history kept with no circular logging. Users must be careful here to manage storage space effectively

Note

If a debug file is specified, it should be opened using either 'w+' or 'r+' before running the `begin()` function of this library.

Example

Simple Specify Debug Output

```
def resetModule():  
    # perform reset of  
    100ms pulse to the RST pin  
    # set reset pulse  
    # wait for 100ms  
    # unset reset pulse  
    pass
```

```
# Creates a new instance  
named 'mates' which  
utilizes:  
# - COM4 as the serial  
port  
# - resetModule as the  
reset function  
MatesController mates =  
MatesController("COM4",  
resetFunction=resetModule)
```

```
def resetModule():  
    # perform reset of  
    100ms pulse to the RST pin  
    # set reset pulse  
    # wait for 100ms  
    # unset reset pulse  
    pass
```

```
# Creates a new instance  
named 'mates' which  
utilizes:  
# - COM7 as the serial  
port  
# - resetModule as the  
reset function  
# - output_file as debug  
file stream  
# - debugFileLength of  
zero indicating no  
circular logging  
MatesController mates =  
MatesController("COM7",  
resetFunction=resetModule,  
debugStream=output_file,  
debugFileLength=0)
```

N

Th
ca

discussion about the methods that
instance.

begin(baudrate)

Begins the serial connection if portname not supplied in constructor.

Parameters	Type	Description
baudrate	str	the baudrate of the serial port (default: 9600)

Return

None

Example

Simple Specify Baudrate

```
# Initializes
display serial port
9600 baud
# and resets the
display if a reset
function is provided
mates.begin()
```

```
# Initializes
display serial port
115200 baud
# and resets the
display if a reset
function is provided
mates.begin(115200)
```

C

Closes opened serial port.

Return

None

Example

```
mates.close() # Closes serial port
```

reset(waitPeriod)

Uses hardware driven signal to hard reset companion device.

Parameters	Type	Description
waitPeriod	int	determines how long to wait (milliseconds) before checking for connection, must be within the uint16 datatype range (default: 5000)

Return

success or failure (*boolean*)

Example

Simple Specify Timeout

```
# Reset the
display and wait
for a period of
5 seconds
(default)
mates.reset()
```

```
# Reset the
display and wait
for a period of
4 seconds
mates.reset(4000)
```

reset(waitPeriod)

Sends a serial command to the connected device to trigger a reset.

Parameters	Type	Description
waitPeriod	int	determines how long to wait (milliseconds) before checking for connection, must be within the uint16 datatype range (default: 5000)

Return

success or failure (*boolean*)

Example

Simple Specify Timeout

```
# Reset the display
and wait for a
period of 5 seconds
(default)
mates.softReset()
```

```
# Reset the display
and wait for a
period of 4 seconds
mates.softReset(4000)
```

setBacklight(backlightValue)

Sets the intensity of the backlight of connected device.

Parameters	Type	Description
backlightValue	int	intensity of backlight, must be between 0 and 15

Return

success or failure (*boolean*)

Example

```
mates.setBacklight(15) # Set backlight value of 15
(max)
```

setPage(pageIndex)

Sets the page to be displayed on the connected device.

Parameters	Type	Description
pageIndex	int	index of page to set as current, must be within the uint16 datatype range

Return

success or failure (*boolean*)

Example

```
mates.setPage(1) # Navigate to Page1
```

getPage()

Returns the index of the current page displayed by the connected device.

Return

Active page index (*int*)

Example

```
activePage = mates.getPage() # Query active page
```

setWidgetValueById(widgetId, value)

Sets the value of a specific widget based on the provided widgetId.

Parameters	Type	Description
widgetId	int	the unique id of the target widget, must be within the int16 datatype range
value	int	the value the target widget will be set to, must be within the int16 datatype range

Return

success or failure (*boolean*)

Example

```
mates.setWidgetValueById(MediaGaugeB0, 50) # Set
value of MediaGaugeB0 to 50
```

getWidgetValueById(widgetId)

Gets the value of a specific widget based on the provided identifier.

Parameters	Type	Description
widgetId	int	the unique id of the target widget, must be within the int16 datatype range

Return

Value of the widget specified by **widgetId** (*int*)

Example

```
# Query the current value of MediaLed4
widgetVal = mates.getWidgetValue(MediaLed4)
```

setWidgetValueByIndex(widgetType, widgetIndex, value)

Sets the value of a specific widget based on the index within a widget type.

Parameters	Type	Description
widgetType	MatesWidget	the type of the target widget
widgetIndex	int	the index of the target widget, must be within the uint8 datatype range
value	int	the value the target widget will be set to, must be within the int16 datatype range

Return

success or failure (*boolean*)

Note

All applicable widget types are listed in [here](#).

Example

```
# Set value of MediaGaugeB0 to 50
mates.setWidgetValue(MATES_MEDIA_GAUGE_B, 0, 50)
```

getWidgetValueByIndex(widgetType, widgetIndex)

Gets the value of a specific widget based on the index within a widget type.

Parameters	Type	Description
widgetType	MatesWidget	the type of the target widget
widgetIndex	int	the index of the target widget, must be within the uint8 datatype range

Return

Value of the widget specified by **widgetType** and **widgetIndex** (*int*)

Note

This function is not applicable to *Int32* and *Float* LedDigits

Example

```
# Query the current value of MediaLed4
widgetVal = mates.getWidgetValue(MATES_MEDIA_LED, 4)
```

setLedDigitsShortValue(widgetIndex, value)

Sets the 16-bit integer value of the Led Digits widget specified by widgetIndex.

Parameters	Type	Description
widgetIndex	int	the index of the LedDigits, must be within the uint8 datatype range
value	int	the value the target LedDigits will be set to, must be within the int16 datatype range

Return

success or failure (*boolean*)

Note

This function is only applicable for *Int16* LedDigits

Example

```
mates.setLedDigitsShortValue(2, 50) # Set value of
LedDigits2 to 50
```

setLedDigitsLongValue(widgetIndex, value)

Sets the 32-bit integer value of the Led Digits widget specified by widgetIndex.

Parameters	Type	Description
widgetIndex	int	the index of the LedDigits, must be within the uint8 datatype range
value	int	the value the target LedDigits will be set to, must be within the int32 datatype range

Return

success or failure (*boolean*)

Note

This function is only applicable for *Int32* LedDigits

Example

```
mates.setLedDigitsLongValue(2, 50) # Set value of  
LedDigits2 to 50
```

setLedDigitsFloatValue(widgetIndex, value):

Sets the 32-bit float value of the Led Digits widget specified by widgetIndex.

Parameters	Type	Description
widgetIndex	int	the index of the LedDigits, must be within the uint8 datatype range
value	int	the value the target LedDigits will be set to, must be within the 32-bit float datatype range

Return

success or failure (*boolean*)

Note

This function is only applicable for *Float* LedDigits

Example

```
mates.setLedDigitsFloatValue(2, 9.989) # Set value of  
LedDigits2 to 9.989
```

setSpectrumValue(spectrumId, gaugeIndex, value)

Sets the value of the column (specified by gaugeIndex) of the spectrum widget (specified by spectrumId).

Parameters	Type	Description
spectrumId	int	the unique id of the Spectrum widget, must be within the int16 datatype range
gaugeIndex	int	the gauge index/column of the Spectrum widget, must be within the uint8 datatype range
value	int	the value the target spectrum column will be set to, must be within the uint8 datatype range

Return

success or failure (*boolean*)

Example

```
# Set value of gauge index 2 of LedSpectrum5 to 64  
mates.setSpectrumValue(MatesLedSpectrum5, 2, 64)
```

setLedSpectrumValue(ledSpectrumIndex, gaugeIndex, value)

Sets the value of the column (specified by gaugeIndex) of the Led Spectrum widget (specified by ledSpectrumIndex).

Parameters	Type	Description
ledSpectrumIndex	int	the index of the LedSpectrum widget, must be within the uint8 datatype range
gaugeIndex	int	the gauge index/column of the Spectrum widget, must be within the uint8 datatype range
value	int	the value the target spectrum column will be set to, must be within the uint8 datatype range

Return

success or failure (*boolean*)

Example

```
# Set value of gauge index 2 of LedSpectrum5 to 64
mates.setLedSpectrumValue(5, 2, 64)
```

setMediaSpectrumValue(mediaIndex, gaugeIndex, value)

Sets the value of the column (specified by gaugeIndex) of the Media Spectrum widget (specified by ledSpectrumIndex).

Parameters	Type	Description
mediaIndex	int	the index of the MediaSpectrum widget, must be within the uint8 datatype range
gaugeIndex	int	the gauge index/column of the Spectrum widget, must be within the uint8 datatype range
value	int	the value the target spectrum column will be set to, must be within the uint8 datatype range

Return

success or failure (*boolean*)

Example

```
mates.setMediaSpectrumValue(4, 3, 48)
# Set value of gauge index 3 of MediaSpectrum4 to 48
```

setWidgetParamById(widgetId, param, value)

Sets the value of a widget parameter based on widget id and parameter id.

Parameters	Type	Description
widgetId	int	the unique id of the target widget, must be within the int16 datatype range
param	int	

Parameters	Type	Description
		the unique id of the target parameter, must be within the int16 datatype range
value	int	the value the target parameter will be set to, must be within the int16 datatype range

Return

success or failure (*boolean*)

Example

```
# Set GaugeA3's Background color to BLACK
mates.setWidgetParamById(GaugeA3,
MATES_GAUGE_A_BG_COLOR, BLACK)
```

getWidgetParamById(widgetId, param)

Gets the value of a widget parameter based on widget id and parameter id.

Parameters	Type	Description
widgetId	int	the unique id of the target widget, must be within the int16 datatype range
param	int	the unique id of the target parameter, must be within the int16 datatype range

Return

The current **param** value of the widget specified by **widgetId** (*int*)

Example

```
# Query the background color of GaugeA3
paramVal = mates.getWidgetParamById(GaugeA3,
MATES_GAUGE_A_BG_COLOR)
```

setWidgetParamByIndex(widgetType, widgetIndex, param, value)

Sets the value of a widget parameter based on widget index and parameter id.

Parameters	Type	Description
widgetType	MatesWidget	the type of the target widget
widgetIndex	int	the index of the target widget, must be within the uint8 datatype range
param	int	the unique id of the target parameter, must be within the int16 datatype range
value	int	the value the target parameter will be set to, must be within the int16 datatype range

Return

success or failure (*boolean*)

Example

```
# Set GaugeA3's Background color to BLACK
mates.setWidgetParamByIndex(MATES_GAUGE_A, 3,
MATES_GAUGE_A_BG_COLOR, BLACK)
```

getWidgetParamByIndex(widgetType, widgetIndex, param)

Gets the value of a widget parameter based on widget index and parameter id.

Parameters	Type	Description
widgetType	MatesWidget	the type of the target widget
widgetIndex	int	the index of the target widget, must be within the uint8 datatype range
param	int	the unique id of the target parameter, must be within the int16 datatype range

Return

The current **param** value of the widget specified by **widgetType** and **widgetIndex** (*int*)

Example

```
# Query the background color of GaugeA3
paramVal = mates.getWidgetParamByIndex(MATES_GAUGE_A,
3, MATES_GAUGE_A_BG_COLOR)
```

clearTextArea(textAreaIndex)

Clears a targeted Text Area.

Parameters	Type	Description
textAreaIndex	int	the index of the target TextArea, must be within the uint16 datatype range

Return

success or failure (*boolean*)

Example

```
mates.clearTextArea(6) # Clear TextArea6
```

updateTextArea(textAreaIndex, textFormat, *formatArgs)

Updates the text displayed within Text Area widget.

Parameters	Type	Description
textAreaIndex	int	the index of the target TextArea, must be within the uint16 datatype range
textFormat	str	the string or text format to be written to the TextArea
formatArgs		zero or more values to be formatted into the provided text format string

Return

success or failure (*boolean*)

Example

Simple Use Text Formatting

```
mates.updateTextArea(2,  
"Mates") # Update  
TextArea2 to "Mates"
```

```
value = 76  
mates.updateTextArea(3,  
"Value is {}", value)  
# Print value to  
TextArea3
```

clearPrintArea(printAreaIndex: int)

Clears a targeted Print Area.

Parameters	Type	Description
printAreaIndex	int	the index of the target PrintArea, must be within the uint16 datatype range

Return

success or failure (*boolean*)

Example

```
mates.clearPrintArea(5) # Clear PrintArea5
```

setPrintAreaColor565(printAreaIndex, rgb565)

Sets the color of a PrintArea Widget based on an rgb565 value.

Parameters	Type	Description
printAreaIndex	int	the index of the target PrintArea, must be within the uint16 datatype range
rgb565	int	16-bit color to set widget to, value must be within uint16 datatype range

Return

success or failure (*boolean*)

Example

```
mates.setPrintAreaColor(4, 0xF800) # Set print color  
of PrintArea4 to RED (0xF800)
```

setPrintAreaColorRGB(printAreaIndex, red, green, blue)

Sets the color of a targeted Print Area.

Parameters	Type	Description
printAreaIndex	int	the index of the target PrintArea, must be within the uint16 datatype range
red	int	

Parameters	Type	Description
		8-bit red component of the target color, value must be within uint8 datatype range
green	int	8-bit green component of the target color, value must be within uint8 datatype range
blue	int	8-bit blue component of the target color, value must be within uint8 datatype range

Return

success or failure (*boolean*)

Example

```
# Set print color of PrintArea7 to GREEN
mates.setPrintAreaColor(7, 0, 255, 0)
```

appendArrayToPrintArea(printAreaIndex, array)

Appends an array of 8-bit integers to a targeted Print Area.

Parameters	Type	Description
printAreaIndex	int	the index of the target PrintArea, must be within the uint16 datatype range
buffer	[int]	the list of values to be appended to PrintArea, values must be within the uint8 datatype range

Return

success or failure (*boolean*)

Example

```
arr = [0xAB, 0xCD, 0xEF]
mates.appendArrayToPrintArea(6, arr) # Append "0xAB,
0xCD, 0xEF" to PrintArea6
```

appendStringToPrintArea(printAreaIndex, textFormat, *formatArgs)

Appends text to a targeted Print Area.

Parameters	Type	Description
printAreaIndex	int	the index of the target PrintArea, must be within the uint16 datatype range
textFormat	str	the string or text format to be appended to the PrintArea
formatArgs		zero or more values to be formatted into the provided text format string

Return

success or failure (*boolean*)

Example

Simple Use Text Formatting

```
mates.appendStringToPrintArea(8,
"Mates") # Append "Mates" to
PrintArea8

value = 108
# Append value as text to
PrintArea9
mates.appendStringToPrintArea(9,
"Value: {}", value)
```

appendToScopeWidget(scopeIndex, buffer)

Appends a list of integers to a Scope widget.

Parameters	Type	Description
scopeIndex	int	the index of the target Scope, must be within the uint8 datatype range
buffer	[int]	the list of values to be appended to Scope, values must be within the int16 datatype range

Return

success or failure (*boolean*)

Example

```
data = {0xF8, 0x7F, 0x1F}
mates.appendToScopeWidget(7, data, 3) # Append data
to Scope Widget 7
```

updateDotMatrixWidget(matrixIndex, textFormat, *formatArgs)

Changes the text displayed by the target Dot Matrix widget.

Parameters	Type	Description
matrixIndex	int	the index of the target DotMatrix, must be within the uint8 datatype range
textFormat	str	the string or text format to be appended to the DotMatrix
formatArgs		zero or more values to be formatted into the provided text format string

Return

success or failure (*boolean*)

Example

Simple Use Text Formatting

```
mates.updateDotMatrix(8,  
"Mates") # Update  
DotMatrix0 to "Mates"
```

```
value = 108  
mates.updateDotMatrix(9,  
"Value: {}", value) #  
Update DotMatrix0 to  
show value
```

getButtonEventCount()

Gets the number of events recorded from applicable button widgets.

Return

Number of recorded button events (*int*)

Example

```
# Get the number of logged button events
buttonEvents = mates.getButtonEventCount()
```

getNextButtonEvent()

Gets the next event source logged from applicable buttons.

Return

Widget ID of the next event button (*int*)

Example

```
# If there is any event recorded
if mates.getButtonEventCount() > 0:
    button = mates.getNextButtonEvent()
    if (button == MediaButton1):
        # if the button pressed is MediaButton1
        # do something
    # add more possible cases here...
```

getSwipeEventCount()

Gets the number of events recorded from swipe gestures.

Return

Number of recorded swipe events (*int*)

Example

```
# Get the number of logged swipe events
swipeEvents = mates.getSwipeEventCount()
```

getNextSwipeEvent()

Gets the next swipe event value.

Return

swipe event (*int*)

Example

```
# If there is any event recorded
if mates.getSwipeEventCount() > 0:
    swipe = mates.getNextSwipeEvent()
    if ((swipe & MATES_SWIPE_SOUTH) ==
        MATES_SWIPE_SOUTH):
        # if swipe is towards from top to bottom
        if ((swipe & MATES_SWIPE_EAST) ==
            MATES_SWIPE_EAST):
            # if swipe is towards from left to right
            if ((swipe & MATES_SWIPE_TLBR) ==
                MATES_SWIPE_TLBR):
                # if swipe is towards from top left to bottom
                right
```

getVersion()

Helper function to obtain the version of the Python Mates Controller library.

Return

Version Information (*str*)

Example

```
# Get the library version number as string
matesVersion = mates.getVersion()
```

getCompatibility()

Helper function to obtain the version of the Mates Studio compatible with this library version.

Return

Compatibility Version Information (*str*)

Example

```
# Get the compatible Mates Studio version number as
string
compatVersion = mates.getCompatibility()
```

printVersion()

Debugging function to print the version of the Mates Studio compatible along with this specific library version.

Return

None

Example

```
# Prints library version and compatible Mates Studio
version to debug serial
mates.printVersion()
```

getError()

This function can be used to investigate errors that occurred while controlling the display module. Description of the possible errors is discussed in [here](#).

Return

Current error code (*MatesError*)

Example

```
# Checks the last error that occurred
error = mates.getError()
if error == MATES_ERROR_NONE:
    # Last command was successful
    pass
```