

PyMata

*A Python Client Abstraction Layer for the
Arduino Firmata Protocol*

Table of Contents

- 1.What is Arduino Firmata?.....1
- 2.What is PyMata.....1
- 3.The PyMata Class Library.....1
 - 3.1.PyMata.....1
 - 3.2.PyMataSerial.....1
 - 3.3.PyMataCommandHandler.....2
- 4.PyMata API.....2

Copyright © November 2013 Alan Yorinks. All Rights Reserved.

1. What is Arduino Firmata?

Arduino Firmata is a high performance hardware abstraction layer that communicates with a host application via the Firmata protocol. Arduino Firmata is packaged as a standard library element of the Arduino integrated development environment (<http://arduino.cc/en/Main/Software>). More detail about Arduino Firmata may be found at: <https://github.com/firmata/arduino> and http://firmata.org/wiki/Main_Page.

2. What is PyMata

PyMata is a Python class library that provides a simple method call abstraction layer for the Firmata protocol. It hides all of the details of the protocol, yet providing all of the functionality and power of Firmata to the client application. It was written to be highly performant and implements the entire current “Standard Firmata” command set. It specifically does not color or modify the data returned by Firmata nor does it put any constraints on the communications to Firmata. PyMata's philosophy is to allow the client application to interpret and control the Arduino with minimal interference.

PyMata was designed to be easily extended and add features to accommodate any feature growth within Arduino Firmata.

3. The PyMata Class Library

The class library consists of three classes, PyMata, PyMataSerial, and PyMataCommandHandler. All code is extensively commented and all method interfaces are well documented in the code.

The client application communicates with PyMata by making method calls to the PyMata class. All other classes provide services to PyMata and are not used directly by the client application.

3.1. *PyMata*

This class is the “main” class of the program. It instantiates and contains the other two classes of the library. It contains the complete API method set and the client application communicates only directly with this class. At startup, PyMata detects the number of digital and analog pins and sizes its internal data structures to match the number of pins discovered. The PyMata class runs in a single thread of execution.

If Firmata introduces a new feature requiring a message to be sent from the client application, a new API method is added to this class.

3.2. *PyMataSerial*

This class provides the data transport mechanism for communication with the Arduino micro-controller. Currently serial communication over USB is supported, but the intention is that as other transport means become available, such as WiFi, this class will be replaced with a class to handle the specific data transport protocol.

This class runs a separate thread dedicated to receiving data from the Arduino. As soon as this thread

detects an available character it places the character into the `command_deque`. The deque will automatically grow when needed to, to assure no data is lost. The deque is thread safe, so there is no blocking required. Using a separate thread assures that data is received as fast as possible.

3.3. *PyMataCommandHandler*

The main function of this class is to process the data placed into the `command_deque` by the `PyMataSerial` class. Data removal and interpretation is performed as a separate thread to help enhance performance.

As each Firmata message is received and interpreted, the appropriate command handler for that command is called. This is accomplished by using a map structure that contains the command as its key and a reference to the command handler method as its associated value.

If a new command is introduced by Firmata, a new method to process the command is written and an entry into the command table is added.

The `PyMataCommandHandler` class also maintains a separate “response” table for digital and analog pins. These tables contain an entry for each pin of the Arduino. Each entry reflects the pin mode, and the last data value reported by Firmata in an Analog or Digital Firmata message (input pins only). These tables use a thread lock to guarantee data integrity since they are shared by both this class and `PyMata`.

4. PyMata API

Documentation for the API is provided in a separate html document, “`pymata_api.html`”, as part of the documentation package.