**Setting up Python controlled Arduino systems**

Reinout Meijboom

Version 0.12 alpha

This is based on the Cronin group work and libraries.

**Note:** if sent with the Arduino Libraries and commanduino-master folders the below is not necessary. Copy all the files in the Arduino Libraries to the folder containing your Arduino Libraries and check if they are available under the sketch → include library tab in the Arduino IDE. After that, simply run the setup.py file in the commanduino-master folder. Then play with some examples.

**Software requirements**

The software requires the following:

* Python 3.6 or higher
* Arduino IDE
* Arduino Command Handler
* Arduino Command Tools
* Commanduino library

1. Install Python from <https://www.python.org/downloads/>
2. Install Arduino IDE from <https://www.arduino.cc/en/Main/Software>
3. Download the repository <https://github.com/Pajables/Arduino-Python>
4. Unzip the contents into a folder called “Arduino-Python” somewhere on your device. Take note of the location.
5. Go into the subfolder called “Arduino Libaries” and copy the folders inside. These are the necessary libraries to run Commanduino on the Arduino. Paste the folders into your Arduino IDE library (default is C:\Program Files (x86)\Arduino\libraries\). This will manually add the libraries to the Arduino IDE. Restart the Arduino IDE if it is open, then check if the libraries are available under sketch → include tab.
6. Commanduino is a Python library for controlling Arduino hardware via a Python interface, rather than hardcoding the behaviour into the Arduino itself. Commanduino acts as a more user friendly front-end for the device. By using the [Arduino-CommandTools](https://github.com/croningp/Arduino-CommandTools) and [Arduino-CommandHandler](https://github.com/croningp/Arduino-CommandHandler) libraries in conjunction with Commanduino, you can essentially control any supported Arduino device through Python.
7. To install Commanduino:
   1. First open a command prompt (Windows) or terminal (Linux)
   2. Note the current working directory (location) is displayed next to the cursor. Change the working directory to the same location that Commanduino is currently saved (within “Arduino-Python) by typing “cd [folder name]” and pressing tab to autocomplete the folder name, then pressing \. Keep going like this until you reach the Commanduino working directory. You have to navigate through the folders in sequence, so it may help to open a file explorer to check the route! To go back a level, type “cd ..”. To change to a different disk, simply type the disk letter followed by a colon, e.g. “d:”
   3. Once you have reached the Commanduino location, type “python setup.py install”. This should automatically carry out the installation if you are in the correct directory and Python has been installed.
8. Work through the included examples!

**Examples:**

1. To initiate an example using Commanduino, first open the Arduino IDE, then go to File -> Examples, and select the example you would like to try.
2. Change the pin definitions to suit your circuit configuration, set the port to the one your Arduino is connected to under Tools -> Ports, taking note of this port.
3. Upload the sketch to the Arduino.
4. Navigate to the Commanduino library folder and go to the relevant example file, which should be under Examples -> commanddevices. Each example has a JSON (Javascript Object Notation) file that contains the configuration information for that example This can be opened with any text editor. In this file make sure that your Arduino port is listed under one of the ports. For Windows you have to add a port definition or change one of the existing ones to “port” : “COMX”, where X is the number of your port.
5. Open the .py file (any text editor should work) for the example and change the directory for the CommandManager.from\_configfile method to match the working directory where the JSON file for the example is located.
6. Run a command prompt or terminal, change directory to the Commanduino example folder, then type python demo.py to run the example.

It may be easier to run python files using a python IDE such as Pycharm Community Edition, which will allow you to make changes and test files quickly.