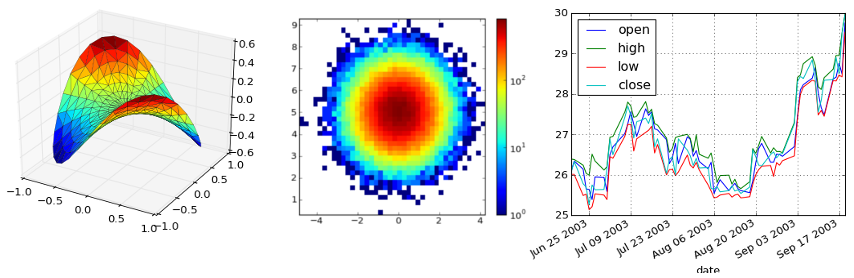
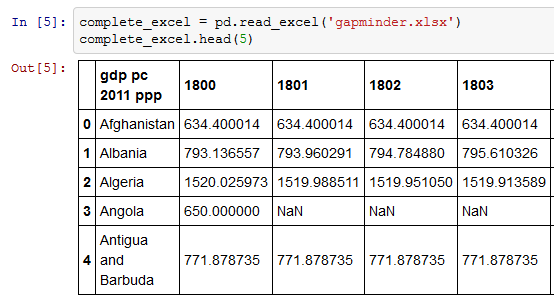
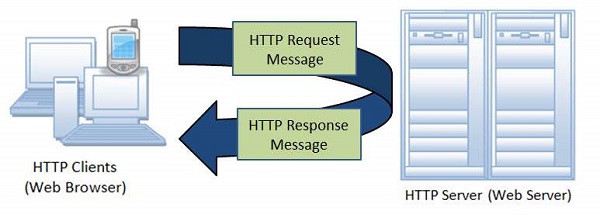
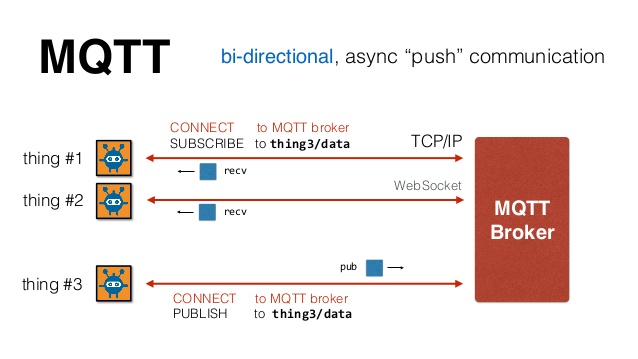
Iot python packages

* **mraa**mraa is a skeleton GPIO library for most SBCs which support Python. The good thing about it is that there's just one library for all devices, so I don't have to use different ones for an Edison and a Pi. Being a high level library, reading from and writing to pins is a one line affair, and the library also provides support for communication protocols such as I2C, UART and SPI.
* **sockets**sockets is a package which facilitates networking over TCP/IP and UDP using Python. It provides access to Berkeley socket APIs to access the Internet. Both TCP/IP and UDP being Transport layer protocols, are ideal for communication with devices on the same WiFi network. One of the more interesting uses of sockets, in my experience is that one can build their own communication protocol using this package as the base.
* **mysqldb**A database is a no-brainer when it comes to most IoT applications. For something whose sole purpose is to send data to the internet, there should be a database, at least a remote one which stores all this generated data. MySQL is the go-to relational database for most developers. In this regard, mysqldb is a very convenient little tool which circumvents the need to execute shell commands within a Python script to read and write to a database.[](https://cdn.instructables.com/FN1/862D/HFSHVH36/FN1862DHFSHVH36.MEDIUM.jpg)
* **numpy**Having used MatLab extensively during my undergraduate studies, I've grown accustomed to dealing with arrays. Python, on the other hand, deals with lists as a substitute for the array which is the same as having a birch tree replace your Rottweiler as the guardian of the house. It just doesn't work. Thankfully, numpy is there to help you out. It is, in essence, a package for scientific computing using Python, very similar to MatLab, but much lighter. The feature I use most is to read sensor data in bulk from my databases and work on them using the inbuilt functions.
* **matplotlib**Data visualization is one of the most fundamental operations that can be performed. It looks pretty impressive when you convert a huge list of numbers to a concise graph which can be understood intuitively. It's also very useful if you happen to be an academician. You know how important those graphs can be in a publication. matplotlib provides a number of different styles of graphs that can be plotted using local data. If you're a data scientist, this is althemore useful to you. Personally, matplotlib is a very useful tool to quickly give me insight to the data I have at my disposal.[](https://itom.bitbucket.io/latest/docs/_images/matplotlib_intro.png)
* **pandas**Another library for data scientists, pandas is a package dedicated towards data analysis. It is in essence, a local alternative to using SQL databases which is more suited to dealing with data as it is built on numpy. It has many advantages over the former, such as a more streamlined approach to data handling and analysis, direct operations on local datasets and the ability to handle heterogeneous and unordered data.[](http://bigdata-madesimple.com/wp-content/uploads/2016/01/Loading-files-into-I-python-notebook-as-a-pandas-dataframe.png)
* **opencv**The big brother of signal processing, image processing, was traditionally the domain of high performance, custom built hardware. Although such devices still carry out the job much faster than their single board counterparts, it is at the very least, a possibility. And, in situations where mobility and connectivity are prioritized over speed, this may just be the solution for those rare times. Opencv is a Python port of the very successful C library for image processing. It contains high-level variants of familiar image processing functions which make photo analysis much easier.
* **tkinter**Although this library does come preinstalled with all installations of Python, its still worthy of a mention. Tkinter is a GUI development library which comes bundled in with all distributions of Python. For people who are more comfortable with a stab wound rather than object oriented programming, learning how to use this package may be a bit daunting at first, but the rewards more than make up for the effort. Every aspect of your Python script can be controlled via a completely ad hoc GUI. This is extremely useful in situations such as functionality testing or repeated executions of the same code.
* **tensorflow**Tensorflow is a package for numerical computations for machine learning. It utilizes a different mathematical representation called data flow graphs which use nodes as mathematical operations and edges as data arrays. This is a very useful library to have if you deal with a lot of non linear datasets or work extensively with decision trees and neural networks.
* **requests**HTTP is one of the major protocols used in traditional internet based resource exchange, being more suited towards large data exchanges. The requests package is used in Python to make HTTP calls and parse responses. This package is useful when dealing with HTTP based third party cloud services.[](https://www.tutorialspoint.com/security_testing/images/http_Protocol.jpg)
* **paho-mqtt**MQTT is a protocol developed solely for for the Internet of Things paradigm. Its focus on high speed communication for low payload communication between resource constrained devices. the paho-mqtt library gives a very user friendly version of the protocol for use with embedded systems. MQTT requests can be made directly within Python, without any additional setup to be done. Especially useful in the prototyping stage.[](https://image.slidesharecdn.com/0xwitjksqnqruoz4tnsi-signature-6a256d24caf5d1fcc6a3bf1d013dfe0e1fa99369a560d140998f50cbdbc6d127-poli-140828123252-phpapp02/95/mqtt-a-practical-protocol-for-the-internet-of-things-9-638.jpg?cb=1409229409)

Which l