Lab 11 –Python / SenseHat (1)[1]

1 Our friend pip

1.1 What is "pip"

Read this entry to get up to speed: https://en.wikipedia.org/wiki/Pip_(package_manager)

Note the part that we may need to use pip3 (it's a Linux thing ☹).

Pip us normally installed with your Python installation. If pip is missing, you will need to install pip yourself or re-install python to get pip set up.

To get more info and help (check it out):

- https://pip.pypa.io/en/stable/

1.2 Updating pip

Issue this command in a shell (dos command shell):

```
python -m pip install --upgrade pip
```

On windows: You will have to have started the dos shell as "administrator". You can do this by left clicking on the shell icon, and then choose to run as an administrator.

On Linux: You probably need to switch "python" with "python3" for this to work[2].

[1] This Lab has not much to do with Sense HAT…
[2] Python 2 is the default python on must Linux distributions (lazy bastards).

2 A Style Guide for Python (PEP8)

2.1 Consistency

It is important to write code in a consistent style. You should obviously write code that is consistent with your own code, and likewise within a project it is essential that people write code adhering to the same style guide. Python does have its own language-wide style guide. You don't have to follow it, but you are generally advised to adhere to it. Or, if you break it, you should have an explicit and convincing explanation for why you did it.

2.2 Python Enhancement Proposal (PEP) 8

So, what's the style guide then?

It is all contained in PEP 8: https://www.python.org/dev/peps/pep-0008/

Read through this document.

- Take note of the various pieces of advice.
- Then start writing code that adheres to the style guide recommendation.

2.3 Style Guide Tool

There used to be a tool called **pep8**, and actually the tool still exists.

- https://pypi.org/project/pep8/

However, the Pythons inventor and Benevolent Dictator For Life (BDFL[3]), Guido van Rossum, did not like the name of the tool. No tool should be named after a PEP.

- https://github.com/PyCQA/pycodestyle/issues/466

So, the package to use is now called pycodestyle.

- https://pypi.org/project/pycodestyle/
- http://pycodestyle.pycqa.org/en/latest/

Install this tool and try it out on some of your Python programs.

[3] However, Guido has now stepped down from the BDFL position.
https://en.wikipedia.org/wiki/Guido_van_Rossum

3 Installation of paho client

3.1 Paho documentation

https://www.eclipse.org/paho/clients/python/

https://www.eclipse.org/paho/clients/python/docs/

3.2 Installation of paho using pip

```
pip install paho-mqtt
```

Please note that on Linux you may need to use **sudo** and on Windows you may need to run the dos command shell as an administrator (right click on dos shell icon, choose “run as administrator”).

It is possible to use virtualenv

4 Minimal test – A Simple Calculator

A very simple distributed calculator (to be used with localhost MQTT server):

We will use the following topics:

- dat235/calc
- dat235/calc/oper/add
- dat235/ calc /oper/sub
- dat235/ calc /oper/mul
- dat235/ calc /oper/div
- dat235/ calc /oper/load
- dat235/ calc /eval

Each of the topics (add, sub, mul, div and load) takes one argument (parameter). The "server" has an accumulator (let's call it accum). The accumulator is published in the **eval** topic.

The interactions are fairly simple.

- The **calculator** will listen to all **oper** sub-topics
    - Initially: accum = 0, and published to the **eval** topic
    - When an **oper** is received (and verified to be add, sub, mul or div), then:
        - Add: accum += nval
        - Sub: accum -= nval
        - Mul: accum = accum * nval
        - Div: accum = accum * nval

      nval is the value in the current message
      nval is expectd to be encoded as a string or a binary value.

    - When **oper/load** is received, then accum = nval, and it is published immediately in the **eval** topic. The value published at **eval** is binary.

The code for the "server" is attached.

The "server" is run in a separate window/shell.

To make life simple, we need only use the "simple" versions.

The user application (CalcApp.py):

- You must write this one 😊 (there is a some sample code attached)
- A small interactive calculator.
- It has functions for each of the operations (add, sub, div, mul) and load.
    - These shall publish to the server and subscribe from evel topic
    - The result of an operation is printed after the operation is run.
- The user uses these functions directly on the REPL interface (interactive python)