# MaxAir Technical – Importing Data From an External Source

### Scenario

I am able to capture the flow and return temperatures from my boiler using a Python script and would like to be able to display these readings in a MaxAir graph. This can be achieved by configuring ‘Dummy’ nodes.

The same technique could be used for displaying data from other external sources.

### Implementation

#### MaxAir

1. A ‘Dummy’ node will be created.
2. A Sensor device will be created and allocated to the ‘Dummy’ node.
3. The Sensor data will be displayed on one of the existing graphs.

#### External System

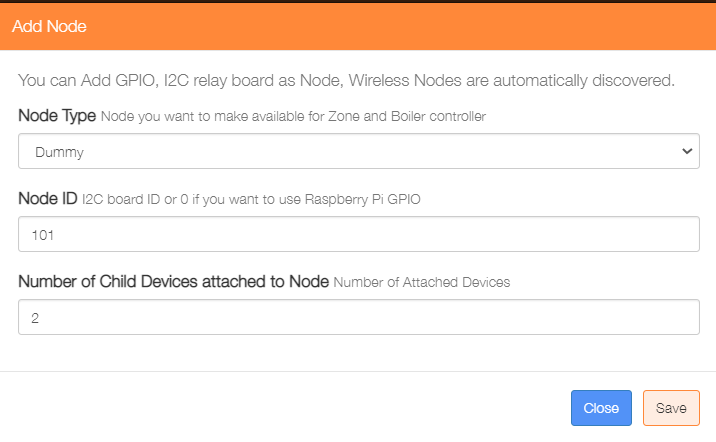
1. The external system will be able to access the MaxAir database from its Python script.
2. The flow and return temperatures will be captured from the boiler.
3. The MaxAir ‘messages\_in’ and ‘zone\_graphs’ tables will be updated using the ‘Dummy’ node IDs created above.

#### MaxAir Configuration



Select Node from the Settings/Node and Zone Configuration menu.

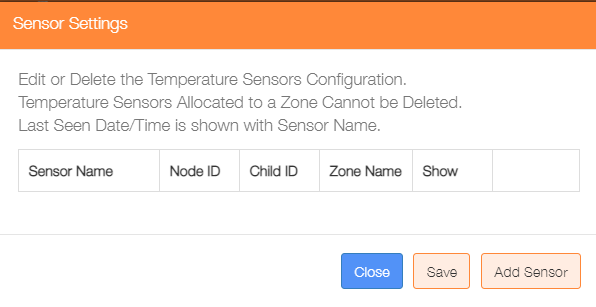
Click on ‘Add Node’.



Add a ‘Dummy’ node type, the ‘Node ID’ can be any value not currently in use, and for this example the ‘Number of Child Devices attached to Node’ will be 2.



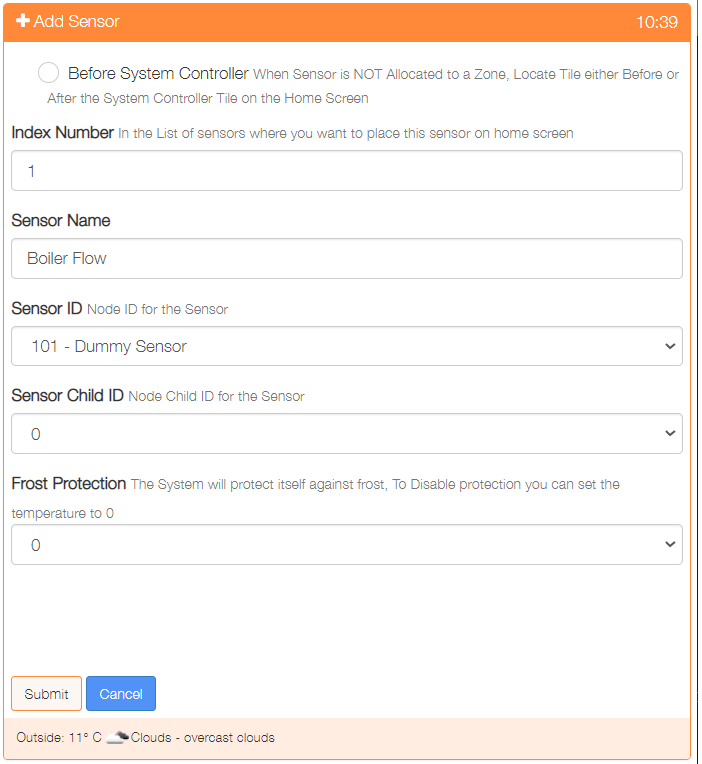
Select the Sensors menu item from the Settings/Node and Zone Configuration menu to display a list of any currently configured sensors.



Click on the ‘Add Sensor’ button to configure the first sensor



An alternative method to go directly to the Add Sensor dialogue, is from the Home screen click on the ‘One Touch’ button then select the ‘Add Sensor’ menu item.

Show either before or after the system controller on the Home screen

Used to order where on the Home screen the sensor is displayed

Provide a name for this sensor device

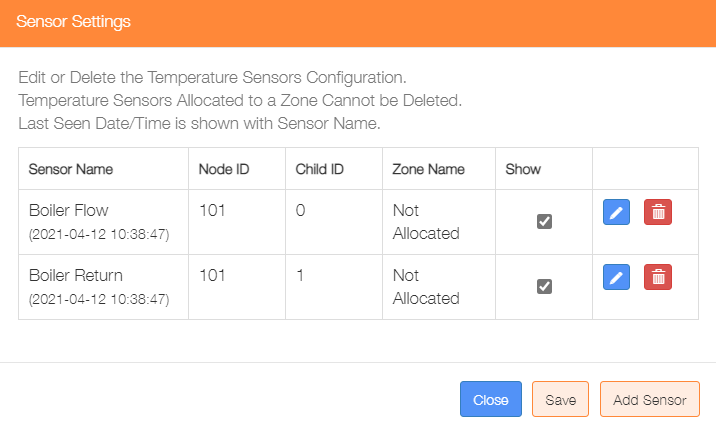
Select the Sensor ID from the dropdown list of available Nodes

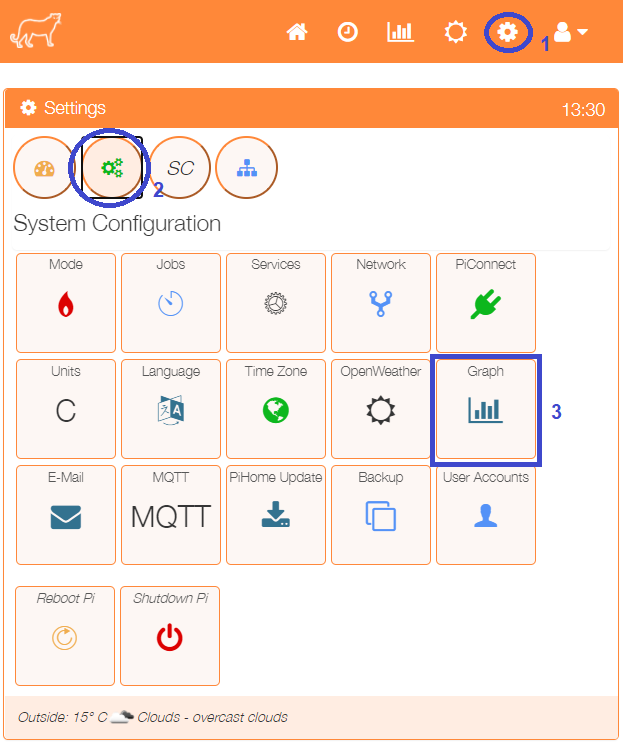
Choose the Child ID from the dropdown list, for nodes with only 1 sensor, this will be 0

Select the frost protection temperature or 0 to disable this feature

Click on ‘Submit’ to add the device.

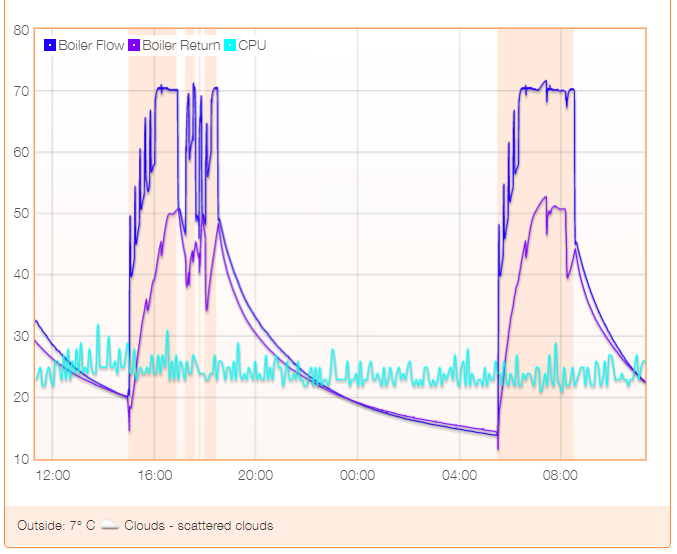
Repeat the process to add any other ‘Return’ temperature.

Re-selecting the Sensors menu item will display the two new sensors. If you wish to show them on the MaxAir home screen, then check the ‘Show’ tickbox/s and then click on the ‘Save’ button.

For this example, the requirement is to provide graphs of the Flow and Return temperatures. Click of the ‘Graph’ menu item from the ‘Settings/System Configuration’ menu.

We will display the data on Graph 3, set the ‘Graph Number’ to 3 for both sensors and click the ‘Save’ button.

The data will be displayed as requested on Graph 3.



#### Example Python Script to Update the MaxAir Database

#!/usr/bin/env python

import time, datetime, MySQLdb

from configparser import ConfigParser

########## Initialise the database access varables ##########

config = ConfigParser()

config.read('/var/www/st\_inc/db\_config.ini')

servername = config.get('db', 'hostname')

username = config.get('db', 'dbusername')

password = config.get('db', 'dbpassword')

dbname = config.get('db', 'dbname')

nodeID = config.get('db', 'kitchen\_node\_id')

########## Initialise the database connection ##########

cnx = MySQLdb.connect(host=servername, user=username, passwd=password, db=dbname)

########## Find the node and child ids for the dummy sensors used to pass data back to the PiHome database ##########

query = ("SELECT \* FROM temperature\_sensors WHERE name = 'Boiler Flow' LIMIT 1;")

cursorselect.execute(query)

results =cursorselect.fetchone()

if cursorselect.rowcount > 0 :

flow\_id = int(results[0])

flow\_sensor\_id = int(results[4])

flow\_sensor\_child\_id = int(results[5])

cursorselect.execute('SELECT node\_id FROM nodes WHERE id = (%s)', (flow\_sensor\_id, ))

results =cursorselect.fetchone()

if cursorselect.rowcount > 0 :

flow\_node\_id = int(results[0])

query = ("SELECT \* FROM temperature\_sensors WHERE name = 'Boiler Return' LIMIT 1;")

cursorselect.execute(query)

results =cursorselect.fetchone()

if cursorselect.rowcount > 0 :

return\_id = int(results[0])

return\_sensor\_id = int(results[4])

return\_sensor\_child\_id = int(results[5])

cursorselect.execute('SELECT node\_id FROM nodes WHERE id = (%s)', (return\_sensor\_id, ))

results =cursorselect.fetchone()

if cursorselect.rowcount > 0 :

return\_node\_id = int(results[0])

Loop reading temperatures from boiler and send to MaxAir

while True:

# Add Flow and Return temperatures to the messages\_in table and update the zone\_graphs table entries

F\_temp = ........ # code here to get Flow temperature from boiler

R\_temp = ........ # code here to get Return temperature from boiler

try :

cursorinsert = cnx.cursor()

cursorinsert.execute('INSERT INTO messages\_in(`sync`, `purge`, `node\_id`, `child\_id`, `sub\_type`, `payload`) VALUES(%s,%s,%s,%s,%s,%s)', (0,0,flow\_node\_id,flow\_sensor\_child\_id,0,F\_temp))

cursorinsert.close()

cnx.commit()

cursorselect = cnx.cursor()

cursorselect.execute('SELECT graph\_num FROM temperature\_sensors WHERE id = (%s)', (flow\_id, ))

results =cursorselect.fetchone()

if cursorselect.rowcount > 0 :

if int(results[0]) > 0 :

cursorinsert = cnx.cursor()

cursorinsert.execute('INSERT INTO zone\_graphs(`sync`, `purge`, `zone\_id`, `name`, `type`, `category`, `node\_id`,`child\_id`, `sub\_type`, `payload`, `datetime`) VALUES(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)', (0,0,flow\_id,"Boiler Flow","Sensor",0,flow\_node\_id,flow\_sensor\_child\_id,0,F\_temp,datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')))

cursorinsert.close()

cnx.commit()

cursorselect.close()

cursordelete = cnx.cursor()

cursordelete.execute('DELETE FROM zone\_graphs WHERE node\_id = (%s) AND child\_id = (%s) AND datetime < CURRENT\_TIMESTAMP - INTERVAL 24 HOUR;', (flow\_node\_id, flow\_sensor\_child\_id))

cursordelete.close()

cnx.commit()

except :

pass

# Add Return temperature to the messages\_in table and update the zone\_graphs table entry

try :

cursorinsert = cnx.cursor()

cursorinsert.execute('INSERT INTO messages\_in(`sync`, `purge`, `node\_id`, `child\_id`, `sub\_type`, `payload`) VALUES(%s,%s,%s,%s,%s,%s)', (0,0,return\_node\_id,return\_sensor\_child\_id,0,R\_temp))

cursorinsert.close()

cnx.commit()

cursorselect = cnx.cursor()

cursorselect.execute('SELECT graph\_num FROM temperature\_sensors WHERE id = (%s)', (flow\_id, ))

results =cursorselect.fetchone()

if cursorselect.rowcount > 0 :

if int(results[0]) > 0 :

cursorinsert = cnx.cursor()

cursorinsert.execute('INSERT INTO zone\_graphs(`sync`, `purge`, `zone\_id`, `name`, `type`, `category`, `node\_id`,`child\_id`, `sub\_type`, `payload`, `datetime`) VALUES(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)', (0,0,return\_id,"Boiler Return","Sensor",0,return\_node\_id,return\_sensor\_child\_id,0,R\_temp,datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')))

cursorinsert.close()

cnx.commit()

cursorselect.close()

cursordelete = cnx.cursor()

cursordelete.execute('DELETE FROM zone\_graphs WHERE node\_id = (%s) AND child\_id = (%s) AND datetime < CURRENT\_TIMESTAMP - INTERVAL 24 HOUR;', (return\_node\_id, return\_sensor\_child\_id))

cursordelete.close()

cnx.commit()

except :

pass

time.sleep(1)