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

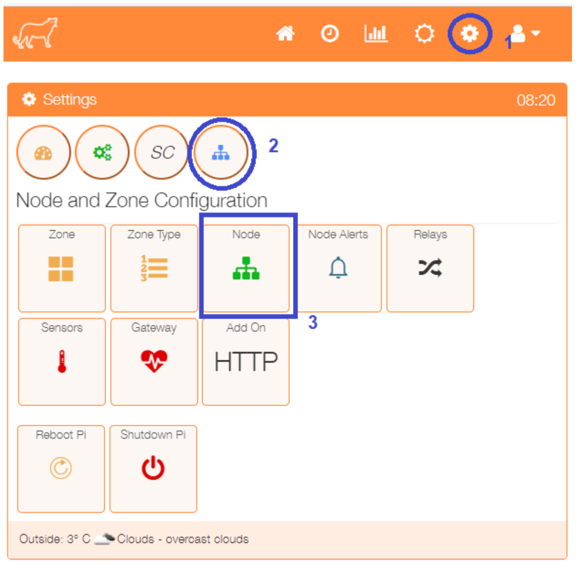
MaxAir

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

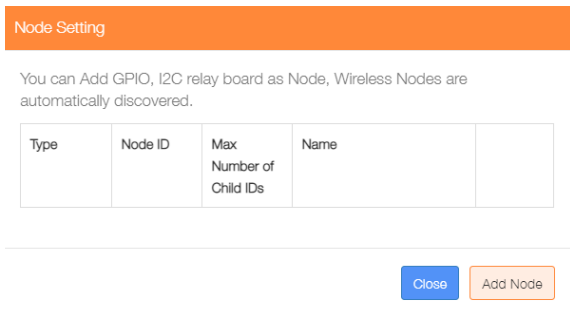
1. The external system will be able to access the MaxAir database from its Python script.
2. The flow and return temperatures will be captured from the boiler.
3. The MaxAir ‘messages_in’ and ‘zone_graphs’ tables will be updated using the ‘Dummy’ node IDs created above.

MaxAir Configuration



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

Click on ‘Add Node’.

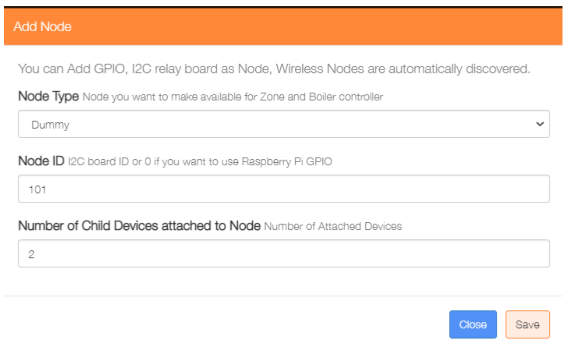


Node Setting

You can Add GPIO, I2C relay board as Node, Wireless Nodes are automatically discovered.

Type	Node ID	Max Number of Child IDs	Name	

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.



Add Node

You can Add GPIO, I2C relay board as Node, Wireless Nodes are automatically discovered.

Node Type Node you want to make available for Zone and Boiler controller

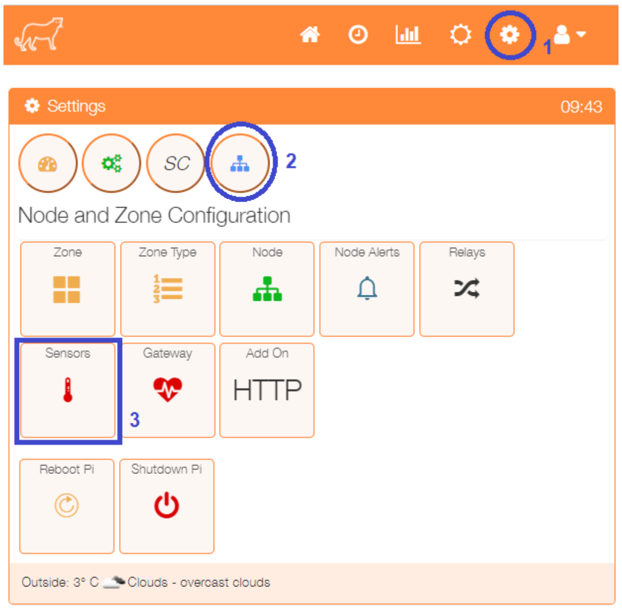
Dummy

Node ID I2C board ID or 0 if you want to use Raspberry Pi GPIO

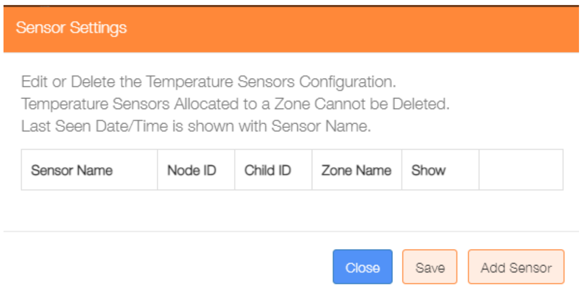
101

Number of Child Devices attached to Node Number of Attached Devices

2



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

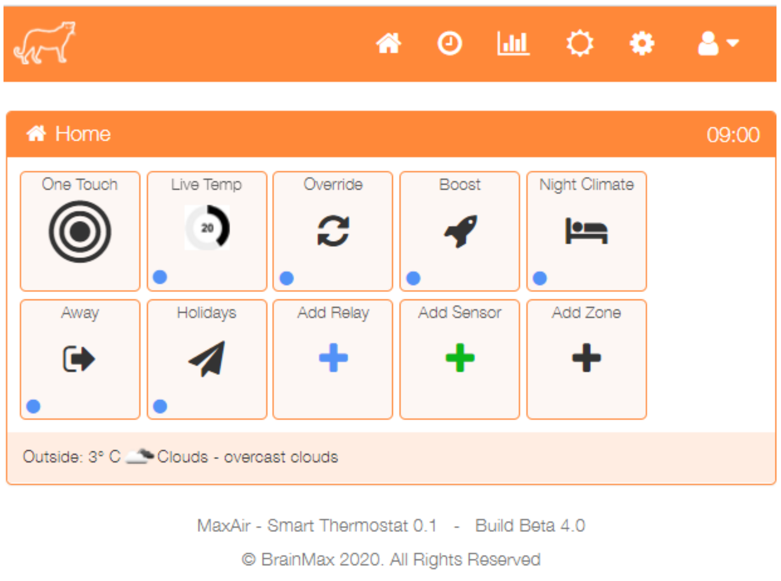

Sensor Settings

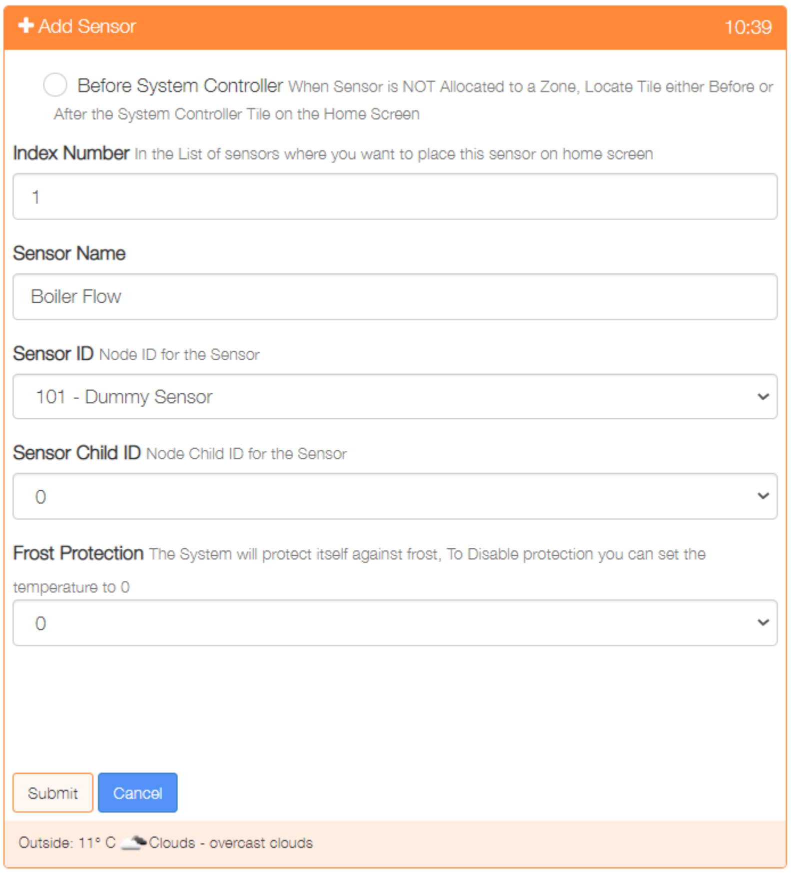
Edit or Delete the Temperature Sensors Configuration.
Temperature Sensors Allocated to a Zone Cannot be Deleted.
Last Seen Date/Time is shown with Sensor Name.

Sensor Name	Node ID	Child ID	Zone Name	Show	

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.


Home
09:00
One Touch
Live Temp
20
Override
Boost
Night Climate
Away
Holidays
Add Relay
Add Sensor
Add Zone
Outside: 3° C Clouds - overcast clouds
MaxAir - Smart Thermostat 0.1 - Build Beta 4.0
© BrainMax 2020. All Rights Reserved



+ Add Sensor 10:39

( ) Before System Controller When Sensor is NOT Allocated to a Zone, Locate Tile either Before or After the System Controller Tile on the Home Screen

Index Number In the List of sensors where you want to place this sensor on home screen: 1

Sensor Name: Boiler Flow

Sensor ID Node ID for the Sensor: 101 - Dummy Sensor

Sensor Child ID Node Child ID for the Sensor: 0

Frost Protection The System will protect itself against frost, To Disable protection you can set the temperature to 0: 0

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

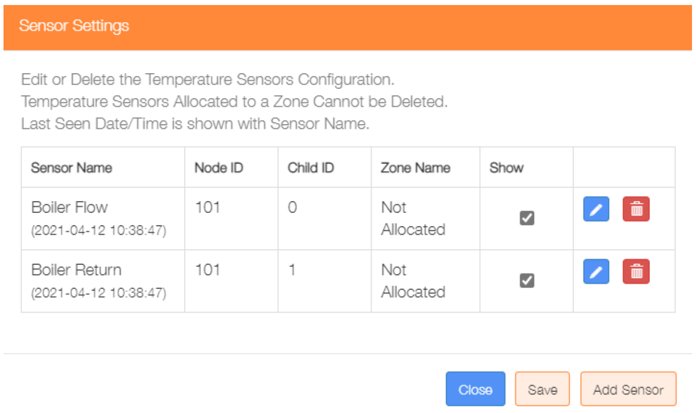
Choose the Child ID from the dropdown list, for nodes with only 1 sensor, this will be 0

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.

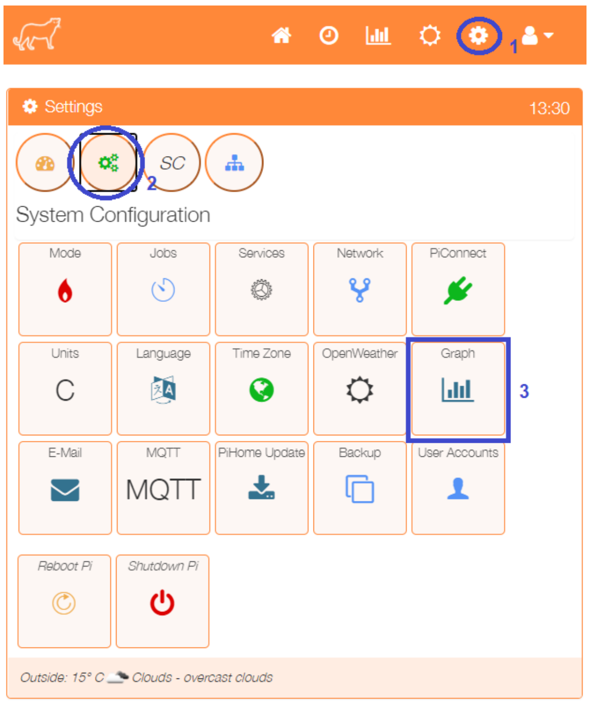
Re-selecting the Sensors menu item will display the two new sensors. If you wish to show them on the MaxAir home screen, then check the 'Show' tickbox/s and then click on the 'Save' button.



Sensor Settings

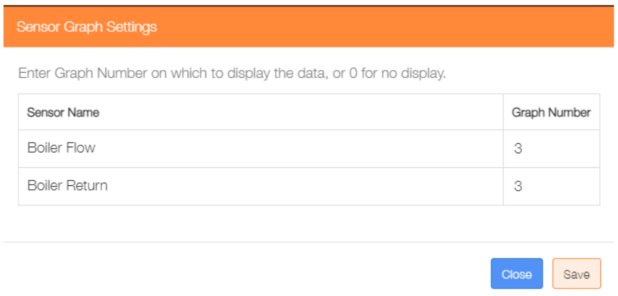
Edit or Delete the Temperature Sensors Configuration.
Temperature Sensors Allocated to a Zone Cannot be Deleted.
Last Seen Date/Time is shown with Sensor Name.

Sensor Name	Node ID	Child ID	Zone Name	Show	
Boiler Flow (2021-04-12 10:38:47)	101	0	Not Allocated	✓	
Boiler Return (2021-04-12 10:38:47)	101	1	Not Allocated	✓	



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.

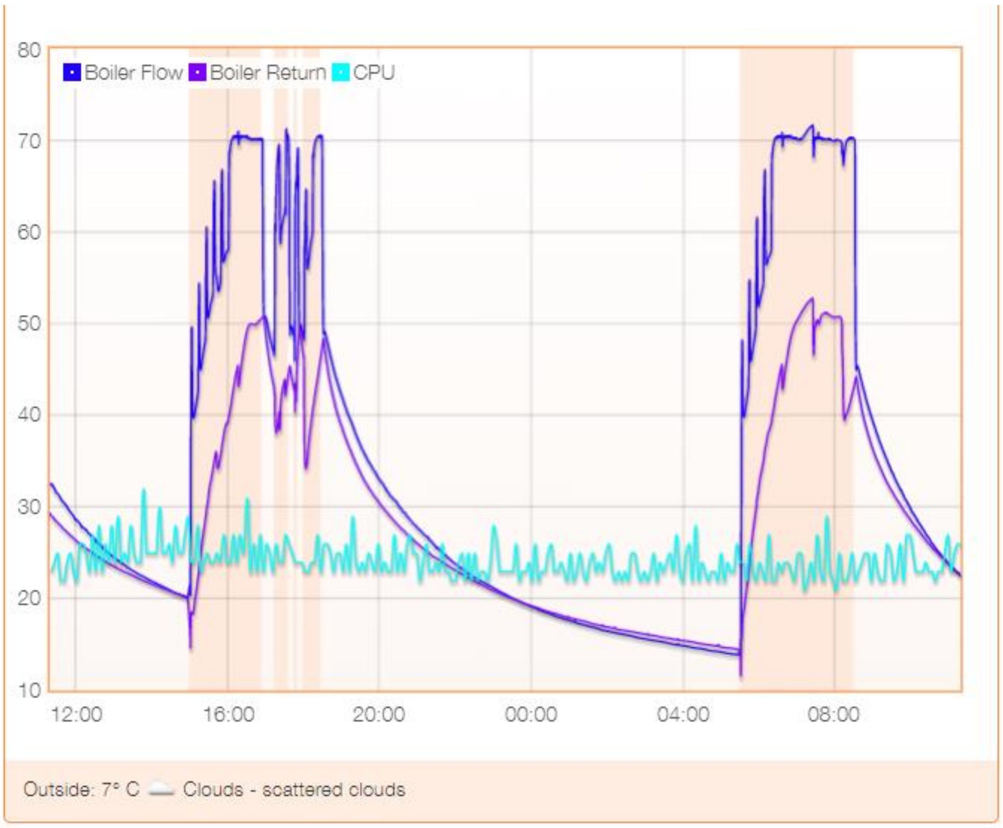


Sensor Graph Settings

Enter Graph Number on which to display the data, or 0 for no display.

Sensor Name	Graph Number
Boiler Flow	3
Boiler Return	3

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)
```