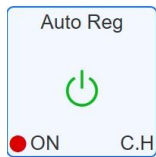


MaxAir Technical – Custom Message Relays

‘Message Relays’ provide the ability to display external text information on a Home Screen relay tile in the lower right text area.



For example, it is possible to display external status information captured from an interface to the boiler. Due to the format of the ‘relays’ table, this information is passed as a numeric code, which must be converted to the message to be displayed on the tile’

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

Implementation

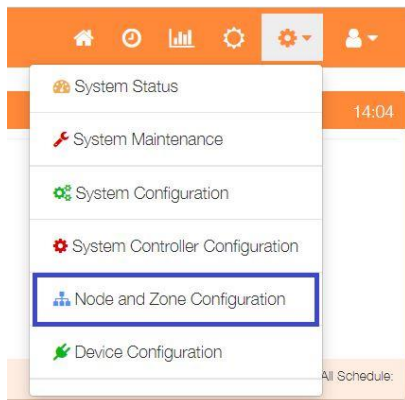
MaxAir

1. A ‘Dummy’ node will be created.
2. A ‘Message Relay’ device will be created and allocated to the ‘Dummy’ node.
3. Mapping information will be created to place the required information on the relay tile.

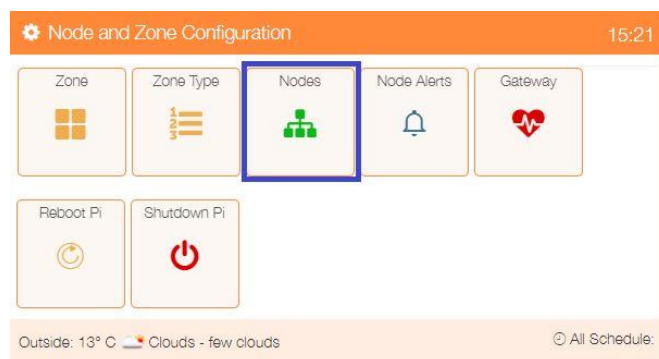
External System

1. The external system will be able to access the MaxAir database from its Python script.
2. The required data in the form of a message code will be captured and used to add an entry to the MaxAir ‘relays’ table ‘current_val_2’ column, using the Relay ID created above.

MaxAir Configuration



Select ‘Node and Zone Configuration’ from the Settings dropdown list, then click the ‘Nodes’ button.



Node Setting

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

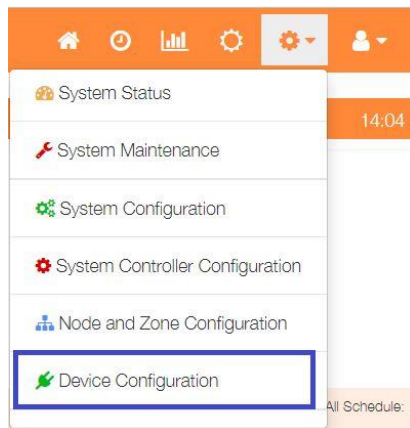
Type	Node ID	Max Number of Child IDs	Name	

Close

Add Node

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 16.



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
 110

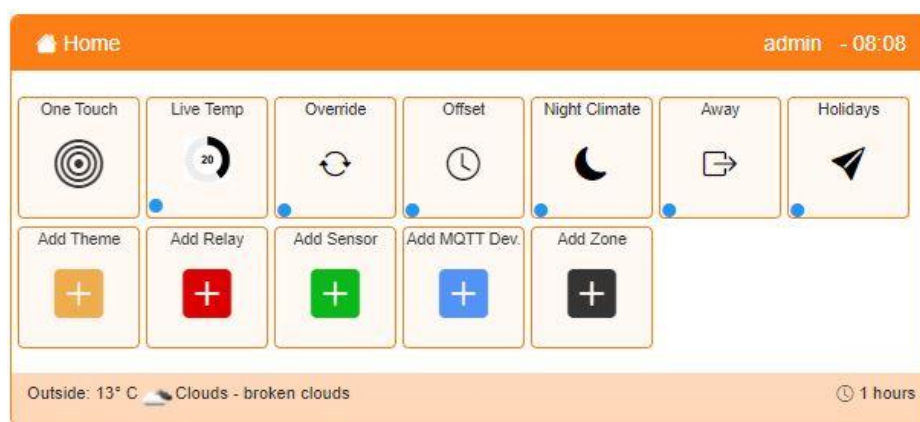
Node Name Identification for the Dummy Device
 Dummy Controller

Number of Child Devices attached to Node Number of Attached Devices
 16

Close Save

Select 'Device Configuration' from the Settings dropdown list, then click the 'Sensors' button.

Click on the 'Add Relay' button to configure the first sensor



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

Add Relay 12:05

☐ Before System Controller When Sensor or Relay 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 Relays where you want to place this Relay on home screen

9

Controller Type (Zone or Pump or Heat or Chill or Fan)

Stand Alone

Relay Name (Identify where the Relay is being used.)

Auto Reg

Relay ID Node ID for the Relay

25 - Dummy Controller

Relay Child ID Node Child ID for the Relay

1

Trigger Level The logic Level to Turn Relay ON.

HIGH

Relay ON Lag Time ON Lag Time in Seconds

0

Fail Timeout Maximum interval in Minutes before the device is considered to have failed to reported. Fault monitoring is disabled for the device if Timeout is set to 0.

0

Group Allocate the Relay to a Group of Relays.

NONE

Submit Cancel

Outside: 11° C Rain - light rain

Show before or after the system controller on the Home screen

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

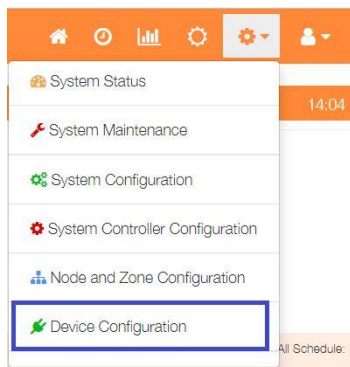
Select 'Stand Alone' as type

Provide a name for this relay device

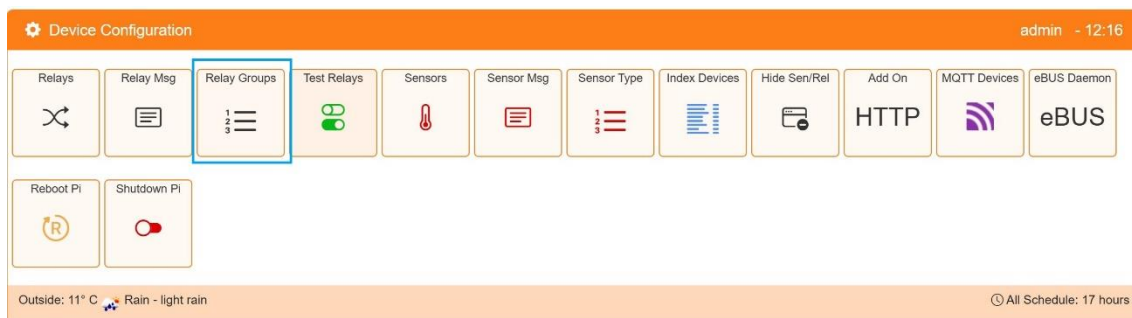
Select the Relay ID from the dropdown list of available Nodes

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

Click on 'Submit' to add the device.



Select 'Device Configuration' from the Settings dropdown list, then click the 'Relay Msg' button.



Custom Relay Messages

Map Message Code to Message Text

Relay	Msg ID	Message

Close Add Msg

To start building the message mapping, click on the 'Add Msg' button.

For a lower right message:

Add Message

Map Message Code to Message Text, and Set Status Icon Colour

Relay Select Message Sensor to which this Message will be attached

Auto Reg

Msg ID Code returned as Sensor Value

5

Message Text Message Text to be displayed

C.H

Close Save

Select the Message Relay

Add the Message numeric code

Enter the text to be displayed

Click 'Save' when completed.

Example Python Script to Update the MaxAir Database

```
#!/usr/bin/env python
import time, datetime, MySQLdb
from configparser import ConfigParser

#### Initialise the database access variables ####
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 ####
con = mdb.connect(dbhost, dbuser, dbpass, dbname)
cur = con.cursor()

# Update MaxAir Database Relays
def update_maxair_relays (conn, relay_id, val_2) :
    cnx = conn.cursor()
    # get 'current_val_2'
    cnx.execute("SELECT * FROM `relays` WHERE `id` = (%s) LIMIT 1;",
        (relay_id,))
    result = cnx.fetchone()
    relay_to_index = dict(
        (d[0], i) for i, d in enumerate(cnx.description)
    )
    relay_name = result[relay_to_index["name"]]
    relay_child_id = int(result[relay_to_index["relay_child_id"]])
    current_val_2 = float(result[relay_to_index["current_val_2"]])
    if val_2 != current_val_2 :
        # update 'current_val_2'
        try :
            query = ("UPDATE `relays` SET `current_val_2` = " + str(val_2) + " WHERE `id` = " + str(relay_id) + ";")
            cnx.execute(query)
            conn.commit()
        except mdb.Error as e:
            print("DB Error %d: %s" % (e.args[0], e.args[1]))
            print(traceback.format_exc())
            logging.error(e)
            logging.info(traceback.format_exc())
            conn.close()
            print(infomsg)
            sys.exit(1)

    cnx.close()
    return True

# Find the node and child ids for the dummy relays used to pass data back to the MaxAir database
# *****
# check if a 'Auto Regulation' relay exists in the database
regulation_relay = False
cur.execute("SELECT * FROM relays WHERE name = 'Auto Reg' LIMIT 1;")
result = cur.fetchone()
if cur.rowcount > 0 :
    relay_to_index = dict(
        (d[0], i) for i, d in enumerate(cur.description)
    )
    auto_reg_id = int(result[relay_to_index["id"]])
```

```

regulation_relay_id = int(result[relay_to_index["relay_id"]])
if int(result[relay_to_index["message_in"]]) == 1 :
    regulation_msg_in = True
else :
    regulation_msg_in = False
cur.execute("SELECT node_id FROM nodes WHERE id = (%s)", (regulation_relay_id, ))
result = cur.fetchone()
if cur.rowcount > 0 :
    node_to_index = dict(
        (d[0], i) for i, d in enumerate(cur.description)
    )
    regulation_node_id = int(result[node_to_index["node_id"]])
    regulation_relay = True

Loop reading status from boiler and send to MaxAir
while True:
    ## Derive the value to pass to the indicate which message to use for the lower right indicator, eg. we will use 'heat_curve'
    as the value ##
    heat_curve = 6
    # set lower right message if dummy relay exists
    if regulation_relay:
        # update 'current_val_2'
        update_maxair_relays(con, auto_reg_id, heat_curve) # Add Error and Current Status to the messages_in table

    time.sleep(1)

```