Table of Contents

[Accessing the PI Computers 1](#_Toc530659153)

[Programs Running at Start-up 1](#_Toc530659154)

[Tail Program 1](#_Toc530659155)

[Eye Program 1](#_Toc530659156)

[Lights Program 1](#_Toc530659157)

[Big LED Program 1](#_Toc530659158)

# Accessing the PI Computers

Under the gold Gromitronic name plate at the front of the dog, there are 2 network ports, one for each raspberry pi.

* Port “One” – connected to the pi controlling the main body LEDs. IP Address is 192.168.0.20
* Port “Two” – connected to the pi controlling the Eyes, Tail and Big LED. IP Address is 192.168.0.10

Passwords for both should be “wensleydale” at all stages.

# Programs Running at Start-up

All the programs are set to run on start-up with a slight delay. We use the crontab file to set these up.

To access the programs that run when the pi starts up, you can type “crontab -e” into the command line. This will bring up the editor for the file, each line is the full path to the file that is being ran at start-up. You can temporarily stop something from running at start-up by putting a ‘#’ at the beginning of the relevant line.

If you need more information on crontab there are many resources online.

# Tail Program

/home/pi/Development/Programs/Wensleydale/TailProgram

The tail program is a python script, RunTail.py. The random wagging can be turned off by setting the boolean “random\_wag” to be false in this file. Note: The tail will still turn off after the set amount of time.

The main part of this script is a try catch block to make sure that the code is stopped in the event of any errors.

The main bulk of the logic for the tail is in HardwareControl/wtail.py

There is also the Main/config.py file that contains the variables for the tail wagging.

# Eye Program

/home/pi/Development/Programs/Wensleydale/EyeProgram

The eye program is a python script, within Scripts/InfiniteScripts, called RunEyes.py

The InfiniteScripts folder contains several different scripts at various levels of randomness in terms of the movements and pauses that the eyes do. RunEyes.py is the most random of these. It also contains the life tests, should these need to be ran again.

Some of the scripts in the scripts folder can be deprecated as the methods are not used in the main infinite scripts. A Deprecated Scripts folder has been made but the individual scripts and methods need checking for whether they are used or not.

The main method to be used for making new movements is the straight\_to\_point method in the wcontroller.py file. This can be used to tell the eyes where you want them to go and how quickly.

The list of positions is in the initialisation for the controller, so any new positions can be added here, and then the range for the random number in RunEyes.py for the position should be increased as well.

If a specific function/movement has been added that cannot be done using the straight\_to\_point loop, in can be added the same way the Gromit\_Eye\_Roll has been used in the RunEyes.py script.

# Lights Program

/home/pi/Development/Programs/Wensleydale/LightsProgram

The lights program is a python script, RunLights.py.

The main part of this script is a try catch block to make sure that the code is stopped in the event of any errors.

There is also the Main/config.py file that contains the variables for the Lights and the pin values.

## Structure

The LED strips are divided up into blocks, some of the LEDs are not visible outside of the dog and so do not belong to any strip.

Patterns can be run separately, but the main program uses the ChangingPattern class. This has been abused slightly as requirements changed, and so is a bit awkward to use. Originally it was only going to deal with button presses and has been tweaked to also take care of the display routine.

## Adding a light pattern.

Patterns are stored in patterns.py, read through the existing patterns to get an idea of how they need to be structured.

Some patterns change every step, some patterns change less often, some change dependant of the WSpeed input.

The pattern needs to set each LED in each block in the input list to the desired colour (or off)

The pattern needs to be added to the WPattern enum.

The pattern needs to be added to the if statement in the update\_blocks function of BlockLightPattern.

The pattern needs to be added to the routine. Either as a new pattern in the button sequences, or as part of the display pattern.

If a display pattern needs to be added, it needs to be added to the update function that looks at the time since the last button press to determine what pattern to show.

If a button pattern needs to be added the corresponding enum should be added to list at the top of the ChangingBlockLightPatterns.init function (config.patternList)

# Big LED Program

/home/pi/Development/Programs/Wensleydale/BigLEDProgram

The big LED program is a python script, RunBigLEDs.py.

The main part of this script is a try catch block to make sure that the code is stopped in the event of any errors.

The main bulk of the logic for the tail is in HardwareControl/wlights.py

There is also the Main/config.py file that contains the variables for the big LEDs and the pin values.

# Logging

# Button Press Counters

The lights pi has a new button press counter function that writes out a cumulative total rather than the recording presses every 10 minutes. The other programs should be updated to use something similar.

# Copying Files via SSH:

Using PuTTY/Tight VNC, find the exact file path to the file you want to copy.

For this to work you will need to know the IP address of the pi you are copying from, and you will need PuTTY installed.

e.g. /home/pi/LightButtonPresses.csv

also find the exact file path to where you want to save the data

e.g. C:\Users\ls141213\Documents

Enter the following command into the windows Command Prompt:

pscp pi@[IP Address]:[PathToFile] [PathToSaveLocation]

following our example, it should look like this:  
pscp pi@169.254.48.0:/home/pi/LightButtonPresses.csv C:\Users\ls141213\Documents

You will be prompted to enter the password for the pi account (it should be wensleydale)

You should then see a progress indicator of the file being copied.

Check in your save location to ensure that the file has copied across okay.

If you get any errors, ensure that PuTTY has been installed correctly, and that pscp is a useable command.

To get a whole folder, after the pscp add -r

# Changing the Date/Time on the Pi

This should be done in PuTTY or through TightVNC

