IoT Rush Challenge

Compete for prizes

Task 1: Control an LED from a web app

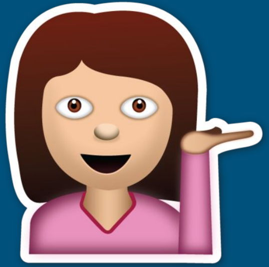
In this task you will assemble and test a simple system which allows a user to control a light from a web-based application. The application is called **‘MyHome’**.

The code for this application is written in **Python** but you don’t need to have prior knowledge of Python to do the task.

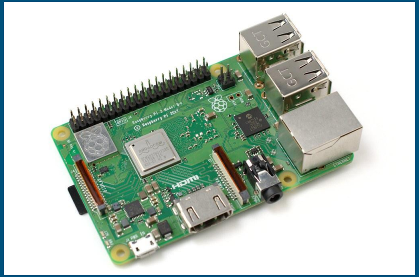
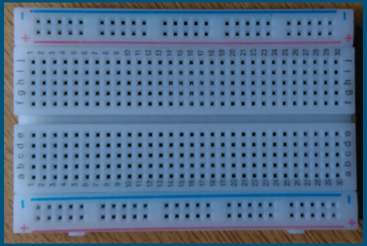
The task has **2 stages:**

A. Build the light circuit and run a simple Python script to test it
B. Compete the web application script and run the app

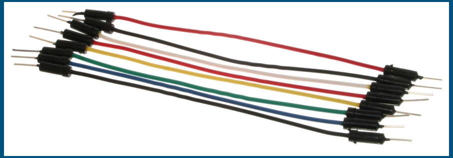
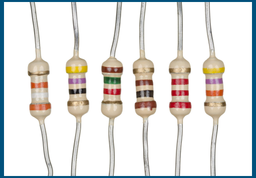
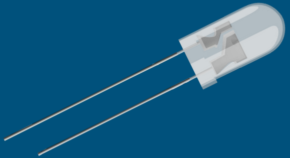
Read **all** the instructions **carefully!**



Check you have the following items before starting:

Raspberry Pi 	A very small computer running a Linux-based operating system. Yours should be up and running already. There should be a folder on the desktop called 'Subject_Focus_Day'. This has the files you'll need today inside it.
Breadboard 	The breadboard is a way of connecting components together in a circuit without having to solder them.

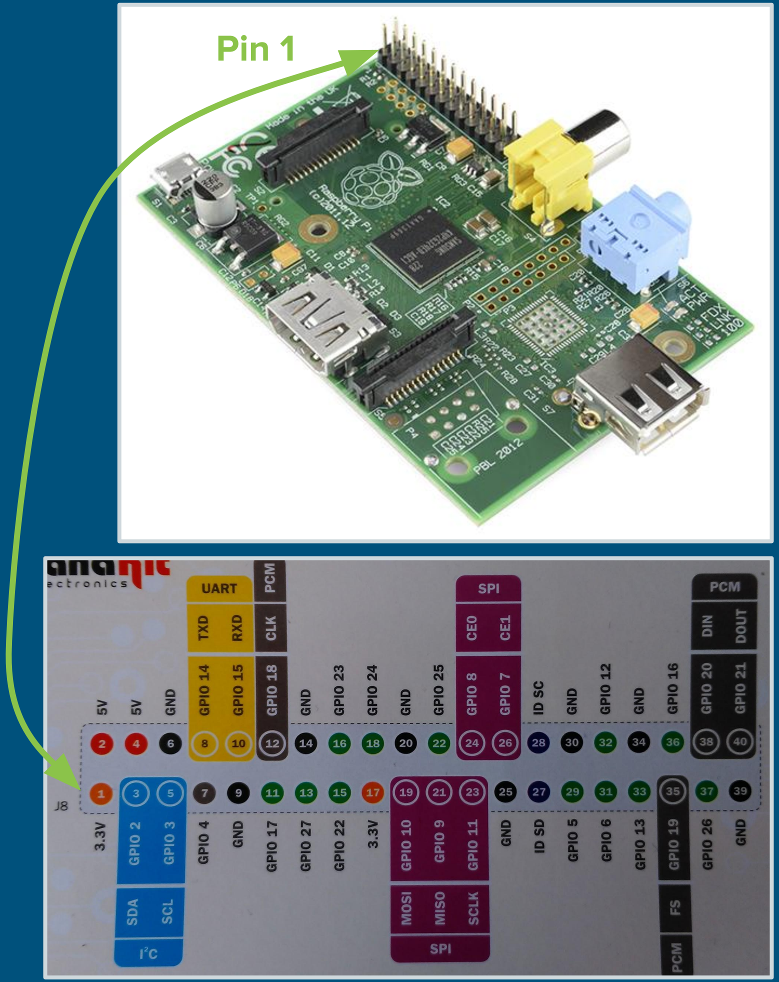
Check you have the following items before starting:

Jumper cables 	These are used to 'jump' from one part of the breadboard to another. You'll need to use the ends with the 'pins' on to connect to the breadboard, and the ends with the holes in will connect to the **GPIO** pins on the Raspberry Pi.
Resistor 	This component controls the amount of current flowing through a circuit. The colours specify the amount of resistance provided by the resistor, which is an amount measured in Ohms.
LED 	A Light Emitting Diode (LED) glows when electricity passes through it. The longer leg (known as the 'anode'), is always connected to the positive supply of the circuit. The shorter leg (known as the 'cathode') is connected to the negative side of the power supply, known as 'ground'.

GPIO Pins

Also worth knowing before you start...

General Purpose Input Output (GPIO) pins provide a way for the Raspberry Pi to interface with the outside world by connecting to external electric circuits. The picture on the right shows the arrangement of the GPIO pins on the model of Raspberry Pi you are using. Pins marked GND are Ground pins.

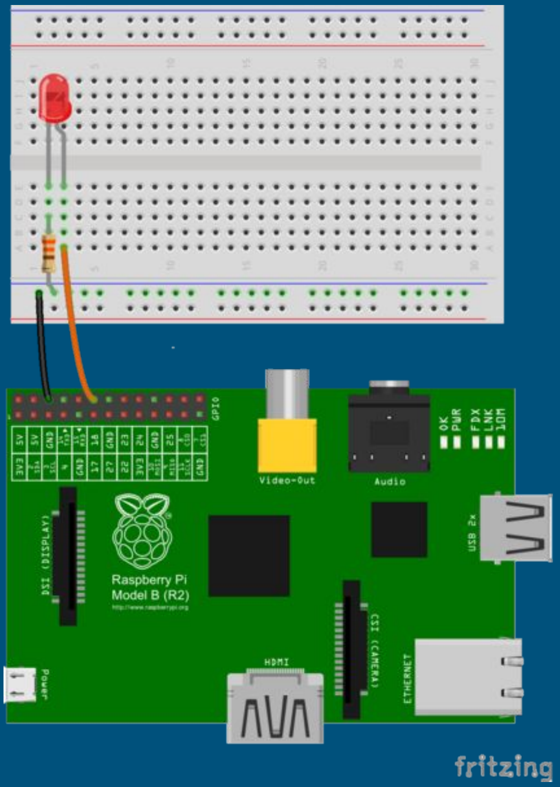


Part A: Build the light circuit

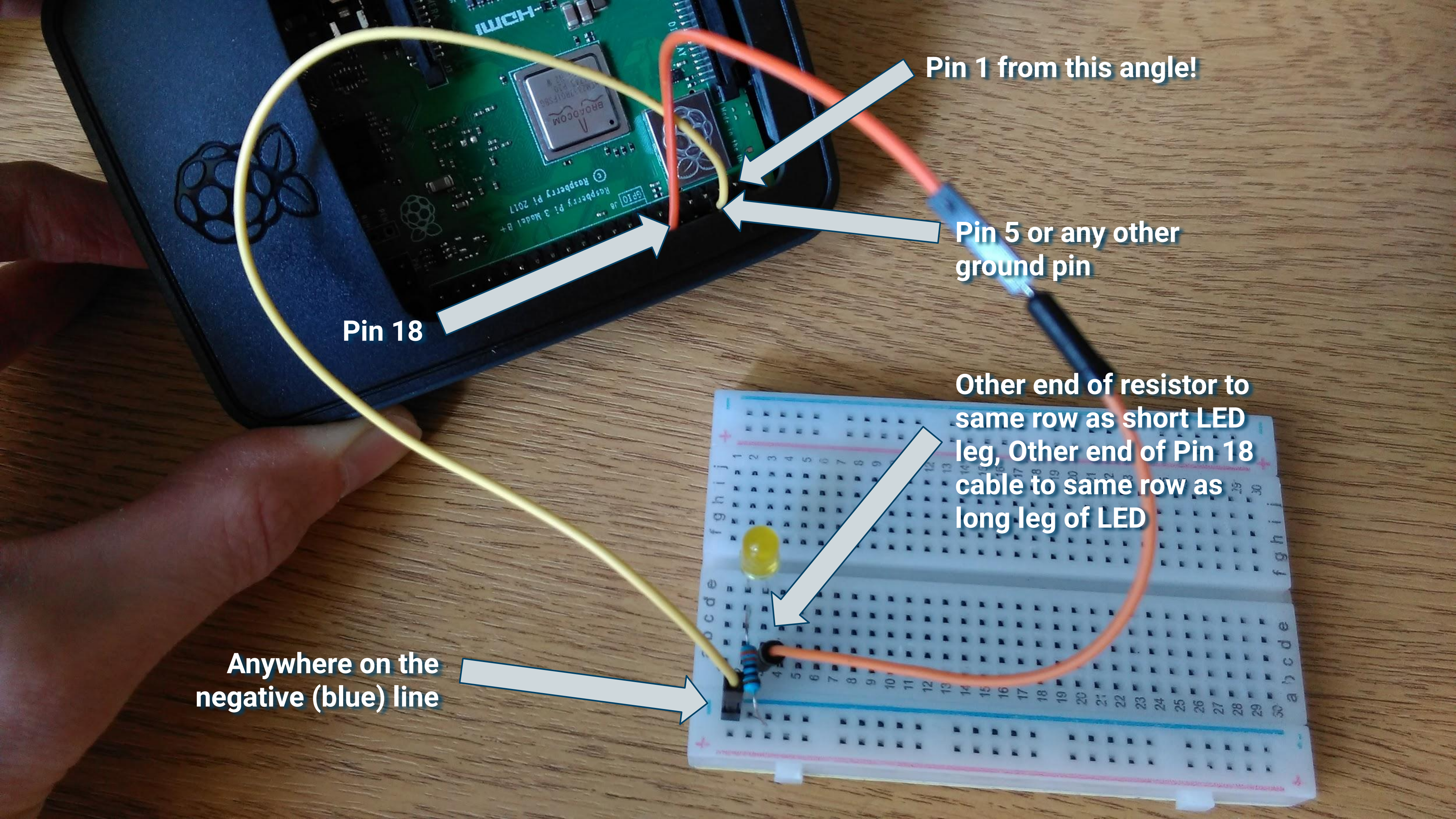
You should turn your Raspberry Pi off for the next bit, just in case you accidentally short something out. You shouldn't need to remove it from its case, but you can if you want to.

- Use one of the jumper wires to connect a ground pin to the rail, marked with blue, on the breadboard. The female end goes on the Raspberry Pi's pin, and the male end goes into a hole on the breadboard.
- Then connect the resistor from the same row on the breadboard to a column on the breadboard, as shown above.
- Next, push the LED legs into the breadboard, with the long leg (with the kink) on the right.

Lastly, complete the circuit by connecting pin 18 to the right hand leg of the LED. This is shown here with the orange wire.



Source: https://thepihut.com/blogs/raspberry-pi-tutorials/27968772-turning-on-an-led-with-your-raspberry-pis-gpio-pins


Pin 1 from this angle!
Pin 5 or any other ground pin
Pin 18
Other end of resistor to same row as short LED leg, Other end of Pin 18 cable to same row as long leg of LED
Anywhere on the negative (blue) line

CanaKit™ electronics

See *http://www.canakit.com/quick-start/pi* for more information

J8

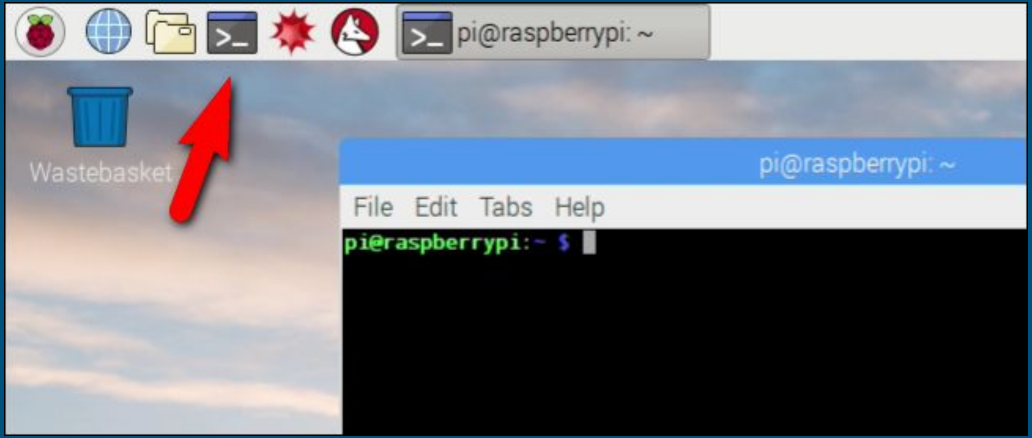
			UART	UART	PCM						SPI	SPI						PCM	PCM
			TXD	RXD	CLK						CE0	CE1						DIN	DOUT
5V	5V	GND	GPIO 14	GPIO 15	GPIO 18	GND	GPIO 23	GPIO 24	GND	GPIO 25	GPIO 8	GPIO 7	ID SC	GND	GPIO 12	GND	GPIO 16	GPIO 20	GPIO 21
2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34	36	38	40
1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39
3.3V	GPIO 2	GPIO 3	GPIO 4	GND	GPIO 17	GPIO 27	GPIO 22	3.3V	GPIO 10	GPIO 9	GPIO 11	GND	ID SD	GPIO 5	GPIO 6	GPIO 13	GPIO 19	GPIO 26	GND
	SDA	SCL							MOSI	MISO	SCLK						FS		
	I²C	I²C							SPI	SPI	SPI						PCM		

Raspberry Pi GPIO Header Quick Reference

Test the light circuit

Next you’re going to run a simple Python script to test your light circuit. The script has been written for you and is called LED.py.

- Click the terminal icon in top toolbar to open a terminal window:



```
pi@raspberrypi:~ $
```

Test the light circuit

- Type the following command to go to the location of the LED.py file in the filesystem:

```
cd ~/Desktop/Subject_Focus_day
```

The ~ symbol is a ‘tilde’ and it is on the right side of the keyboard above the Shift key

- Pressing <Enter> on the keyboard will run the command.
- Next run the following command:

```
python LED.py
```

This command told the Python interpreter to run the LED.py script.

Test the light circuit

Did your LED light up?

Ask a tutor if you ran into problems.

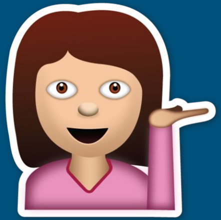
Part B: Control the light from a web app

Next you need to complete the web application so the light can be controlled from a web browser.

Run the web application

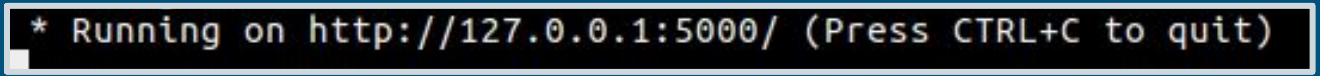
- From the same terminal, type the following commands:

```
source bin/activate
cd app
python routes.py
```



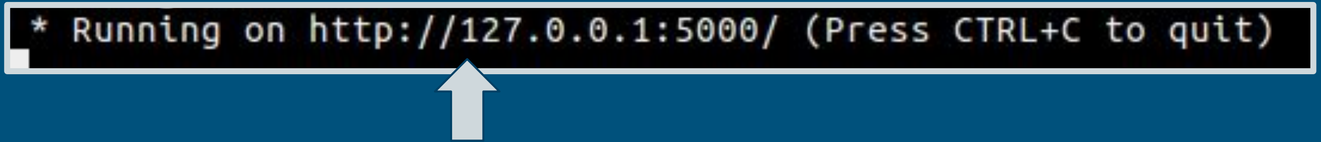
Ask a question if you want any of this explaining!

You should now see a message in the terminal like this one:



```
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Access the app in a browser



```
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

- <CTRL+Click> on the **link** and the app should open in a browser.

See anything?

Clicking the button won’t do a thing of course because we haven’t put in the right code yet!

Edit the app code

There are **2 Python files of interest** inside the **Subject_Focus_Day** folder on your desktop: **LED.py** and **app/routes.py**.

LED.py is the script we ran to test the light circuit; routes.py is the script for the web app.

- **Right click** the LED.py file and select, ‘Open with > Python 3 (IDLE)’
- **Identify 2 lines of code which you think are responsible for turning the light on and off**. A comprehensive explanation of the code is available via the link on page 6.
- **Locate routes.py** (inside the app folder) and open it in Python 3 (IDLE).
- **Copy and paste the 2 lines from LED.py to the appropriate lines in routes.py file**. Comments (in red) offer some help. Note that the lines you insert must be indented to the same level as the comments. These lines of code will be executed in response to different URLs being requested by the browser. For example, http://127.0.0.1:5000/on will trigger the ‘on’ code, while http://127.0.0.1:5000/off and http://127.0.0.1:5000/ will trigger the ‘off’ code.
- **Save** the file (<CTRL+s>)

Restart the app

- **Stop and restart** the web app from the terminal window:

```
<CTRL+c>
python routes.py
```

- **Refresh** the page in the web browser (F5)
- **Try clicking** the button

What happened?

If the light went on/off, you're done with Task 1!

Task 2: Design the perfect user interface

In this task you will improve the design of the existing ‘MyHome’ app interface.

The task involves a little bit of CSS. CSS is short for ‘Cascading Style Sheets’ and it is a markup language used to style web pages.
Don’t worry if you haven’t learnt CSS before – these instructions should help you make some basic changes to the appearance of the ‘MyHome’ interface.

First some basic design principles!

Before you edit the CSS to change the look of the page, here are a few tips for designing user-friendly buttons...

- Make your buttons look and behave like buttons
- Label the buttons with what they do for users
- Avoid relying on colour alone to convey important information
- Avoid certain colour combinations (e.g. red and green)

Keep these in mind when considering what edits to make to the CSS.

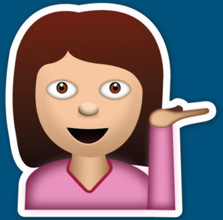
Edit the CSS

- Inside the **Subject_Focus_Day** folder, locate the '**style.css**' file (it's in **app** > **static** > **css**)
- Right click the file and do **'Open with** > **Text Editor'**

Notice that this file contains some blocks of 'code'.
Each one of these blocks targets some specific element(s) in the page and applies some style rules to it.

For example, this block targets the button identified by the 'lightOff' id and makes its text red:

```
a#lightOff {
        color: red;
}
```



Ask a question if you want any of this explaining!

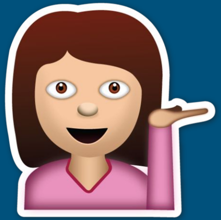
Edit the CSS

Here are a couple of other style rules you could experiment with:

```
    box-shadow: 2px 2px;
    background-color: #DDD;
```

You can also target an element in a particular `state', like when a user **hovers** over it:

```
a.button:hover {
    background-color: #CCC;
}
```



Ask a question if you want me to explain this.

Edit the CSS

There are more examples of things you can do with CSS here:
https://www.w3schools.com/css/default.asp

And here is a neat tool for identifying colours as hexidecimal codes:
https://www.google.com/search?q=color+picker

Have a play!

Task 3: Plan a pitch

Imagine you are seeking investment to develop a marketable version of the ‘MyHome’ app.

For this final task, plan a **2-minute** pitch to an audience of potential investors.

Because 2 minutes is very little time, select some ‘key points’ and communicate them clearly.

Suggested points:

- What will your app do?
- Who is your app aimed at?
- How will they benefit from using it?
- What is novel about it?

Ready to pitch?