Getting started with Node-RED

Node-RED is a drag-and-drop visual tool which comes pre-installed on Raspbian. In this resource we will use Node-RED to control LEDs via the Raspberry Pi's GPIO pins.

One powerful feature of the Raspberry Pi is the row of GPIO pins along the top edge of the board. GPIO stands for General-Purpose Input/Output. These pins are a physical interface between the Raspberry Pi and the outside world. At the simplest level, you can think of them as switches that you can turn on or off (input), or that the Raspberry Pi can turn on or off (output).

The GPIO pins are used to connect the Raspberry Pi to electronic circuits, which allow it to control and monitor the outside world. Using the GPIO pins, the Raspberry Pi is able to turn LEDs on or off, run motors, and perform many other actions. It is also able to detect external conditions, such as temperature, light levels, and whether a switch has been pressed. We refer to this as physical computing.

There are 40 pins on the Raspberry Pi (26 pins on early models), and they provide various different functions.

If you have a RasPiO pin label, it can help to identify what each pin is used for. Make sure your pin label is placed with the keyring hole facing the USB ports, pointed outwards.
If you don't have a pin label, then this guide can help you to identify the pin numbers:

![Image of Raspberry Pi with pin numbers]

You'll see pins labelled as 3V3, 5V, GND and GP2, GP3, and so on:

<table>
<thead>
<tr>
<th>Pin</th>
<th>Voltage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3V3</td>
<td>3.3 volts</td>
<td>Anything connected to these pins will always receive 3.3V of power</td>
</tr>
<tr>
<td>5V</td>
<td>5 volts</td>
<td>Anything connected to these pins will always receive 5V of power</td>
</tr>
<tr>
<td>GND</td>
<td>ground</td>
<td>Zero volts, used to complete a circuit</td>
</tr>
</tbody>
</table>
These pins are for general-purpose use and can be configured as input or output pins.

**WARNING**: if you follow the instructions, then playing about with the GPIO pins is safe and fun. Randomly plugging wires and power sources into your Raspberry Pi, however, may destroy it, especially if you are using the 5V pins.

Wire up the LED

1. Wire up an LED to GPIO pin 17 on your Raspberry Pi by following this diagram:
The positive leg of the LED is usually longer, and it is this leg which should be inserted into the left side of the breadboard (e1 on the diagram).

1. Start up your Raspberry Pi. Click on the Raspberry icon, then the **Programming** menu to open Node-RED.
2. You should see a window displaying information about Node-RED starting up.
3. Now go to the **Internet** menu and open **Chromium Web Browser**.

![Internet menu](image)

4. In Chromium, locate the address bar at the top and type in `localhost:1880`, then press Enter. This will display the Node-RED interface. (Your Raspberry Pi does not need to be connected to the internet to use Node-RED: `localhost` is the address the Raspberry Pi uses to refer to itself and `:1880` means that it is looking at port 1880.)
Connecting to a GPIO pin

Programs in Node-RED are called **flows**. You can see that your blank page is labelled as **Flow 1** in the tab at the top. You can create as many flows as you want and they can all run at the same time. For this guide, we will only need one flow.

1. The coloured blocks on the left side of the interface are the **nodes**. Scroll right down to the bottom of the list and you will see some nodes labelled **Raspberry Pi**.
2. You will see two nodes with the label rpi gpio: these are the ones we will use to talk to the GPIO pins on the Raspberry Pi. The first one in the list, with the raspberry icon on the left, is for inputs. Using a button push to control something would be an example of an input. The second node, with the raspberry icon on the right, is for outputs. Switching on an LED would be an example of an output. Drag an output node onto the blank page in the middle.

![Node configuration](image)

3. Double-click on the node and a box will appear to let you configure the node. Change the GPIO pin to be GPIO17 and tick Initialise pin state?. Leave the setting for Initial level of pin on low. Give the node a name - we called it Green LED because the LED we used was green, but if yours is a different colour feel free to change the name. When you are finished, click Done.
1. Now scroll back up to the list of nodes. To turn the LED on and off, we need an input. In Node-RED we can inject messages into the flow and cause things to happen as a result. Drag an inject node onto the flow.

2. Double-click on the inject node. Use the drop down next to Payload to change the data type to string and type 1 in the Payload box - this will be the message. Type On in the Name box. Press Done.
3. Repeat the previous steps to create another inject node, except this time add 0 as the payload message, and call this node Off.

4. Now look for the grey dot on the right side of the inject nodes. Click and drag from the grey dot on the On node to the grey dot on your LED node to join them up. Repeat for
the **Off** node, also joining it to the LED node.

1. Our flow is finished, so we can deploy it. Click on the big red **Deploy** button on the top right of the screen. A message should pop up at the top saying "Successfully deployed". This is similar to pressing the green flag in Scratch or F5 to run your code in Python.

2. Now click on the blue square on the left of the **On** node to inject the message 1. The **Green LED** node receives the message and your LED should light up. You should be able to turn the LED off by clicking the blue square on the **Off** node, which injects the message 0.
Debugging your flow

1. If your LED doesn't turn on and off, firstly check that you have wired the components correctly on the breadboard. Make sure you have wired your LED to both **ground** and **pin 17** on your Raspberry Pi.

You can also ask Node-RED to display debugging information by wiring up your nodes to a **Debug** node, which can be found under **output**. Drag in a debug node and wire your two inject nodes to it, then click Deploy. When you click the buttons to inject the message, Node-RED will show you what was injected in the **Debug** tab on the right side of the screen. Click the tab to display the messages.

For example, this is the output if you click first on the **On** node and then on the **Off**
1. Now let's add a button to control the LED. Wire up your button to the Raspberry Pi as shown in the image below, so that your LED is still connected to GPIO pin 17, and your button is connected to GPIO pin 4:

![Image showing button wiring](image)

2. Remove your **Turn on** and **Turn off** inject nodes by clicking the node and pressing **Delete** on the keyboard. We no longer need these, as we will be controlling the LED using a physical button.

3. Now we need to add a Raspberry Pi GPIO input node. This is the node with the raspberry icon on the left. Set up this node to receive input from your physical button.
as follows:

Specifying **pullup** means that GPIO pin 4 will be set to **HIGH**, and pressing the button will cause it to go **LOW**.

4. Now join up your button node output to the existing debug and LED nodes. Deploy the flow and test it by pressing the button.

You will notice that the LED is lit to start with, and that pressing the button switches it off. That's not quite right! This is because we are using **pullup** as explained in the previous step, so the button pin will be **HIGH** by default. **HIGH** generates the message 1, and this turns the LED on. When we press the button, we cause the pin to go **LOW**, generating a 0 message which turns off the LED. We need to reverse the values. We want the LED to receive the message 0 by default and the message 1
when the button is pressed.

5. Remove the connection you made between the button node and the LED node by clicking on the line and pressing **Delete** on the keyboard.

6. Add a switch node. This can be found in the **function** section. This node is similar to the **if / elif / else** type constructs you may have seen in Scratch or Python. You can configure it to have multiple output paths (outlined in red on the screenshot) depending on the value passed in. In this case we will set up the node so that if the property **msg.payload** is equal to 1, the first path will be followed. Click the small **Add** button at the bottom to add a second path, and for this path select **otherwise** in the drop down. This path will be followed if the input was anything other than 1. Click **Done** when you are finished.
7. You should see your newly created switch node, with two dots on the right side for outputs. Note that the title **If input is 1** is simply a description of what the node does, and has no effect on its function.

![Switch Node](image)

8. Join up the GPIO input button node to the input (left side) of the switch node.

9. Now drag in a yellow **change** node from the functions section and double-click on it to configure it. We will use this node to change the message being sent. Remember: when we created the switch node, the first output was set to be followed if the input message was 1. We will use the change node to change the message to 0.

![Change Node Configuration](image)

10. Press **Done**, then draw a line from first output of the switch node to the change node. Then connect the output of the change node to the LED node:
11. Now add another change node to set the `msg.payload` to 1. Connect this node to output 2 of the switch node and then to the LED node. This tells the flow that when the `otherwise` branch is followed (i.e. the `msg.payload` is not already 1), we would like to change it to 1. When you are ready, deploy your flow, and then push the physical button to confirm that it works properly.

What next?

Now that you have a single LED working, why not try wiring up two more LEDs to different pins on your Raspberry Pi, and creating a traffic light simulator? Can it be controlled with a button?

To do this, you will need to use more of the nodes from the function section. The delay node allows you to wait for a given number of seconds.

For all nodes, you can drag them in as you did before, and double-click on them to change their configuration. Don't forget that you can link a node to more than one other node. For example, the Start node below is linked to both the Red LED and the delay 5 s nodes. Use this example to get you started: