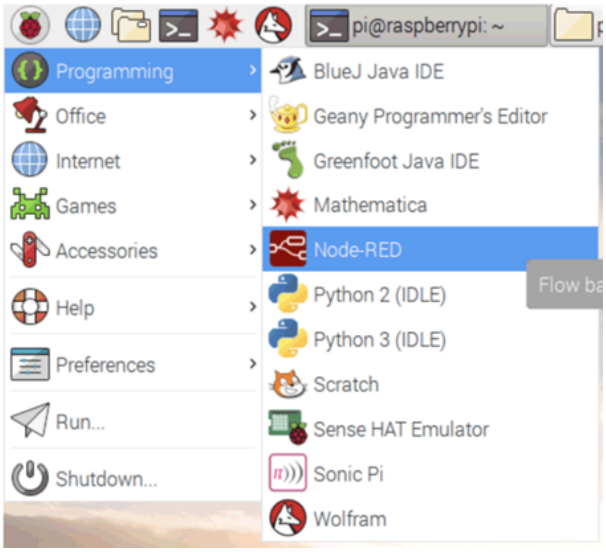
7.Node RED: Connect LED to Internet of Things

Remark: Signature:

Node-Red can be launched via the menu section from your raspberry pi's desktop, via the terminal, or over ssh.

To launch on the Raspberry Pi's desktop, Click on the Raspberry icon, hover your mouse on Programming, click on Node-RED (**Menu>Programming>NodeRed**) to launch it.



It can also be launched from ssh or terminal by running;

```
node-red-start
```

You should see a window like the one below showing node red launch on the desktop.

Once you see this, **go to *menu->internet* and launch the chromium web browser**. While your Raspberry pi does not need the internet to run Node-Red, it uses a browser as its interface.

With chromium launched enter ***localhost:1880*** in the address bar followed by the enter key. 1880 is the port on the raspberry pi on which Node-Red is preset to communicate. This should display the **Node-Red interface** as shown in the image below.

Understanding the Node-RED interface

The **Node-Red interface** comprises the **flow panel, the nodes palette, the debug console and the info console** as highlighted in the image above.

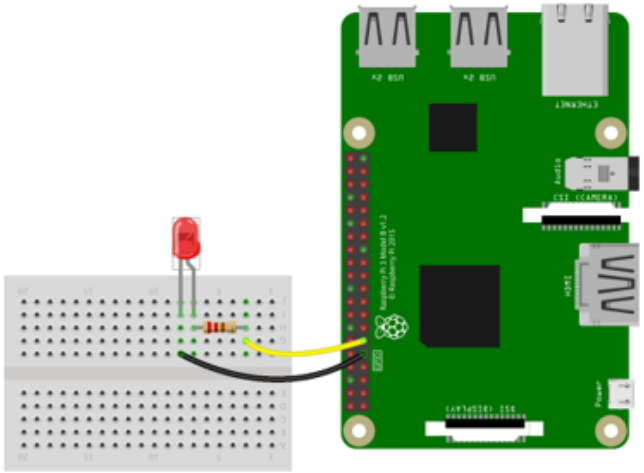
The **flow panel** is where the nodes are joined together to create a program which in Node-Red is called flow, while the **Node palette** comprises objects which are representative of hardware, protocols and software features associated with devices. It includes protocols like MQTT for IoT, and GPIO output and input modes for boards like the raspberry pi. The **info console** provides information on highlighted/selected objects while the **debug console** works just like the Arduino Serial monitor and can provide feedback while the flow is running. The deploy button is used to upload the flow to target hardware. Menu button contains different upload types to help you get the best out of your project. With Node-Red up and running, we can now proceed to build the demo project.

Schematics

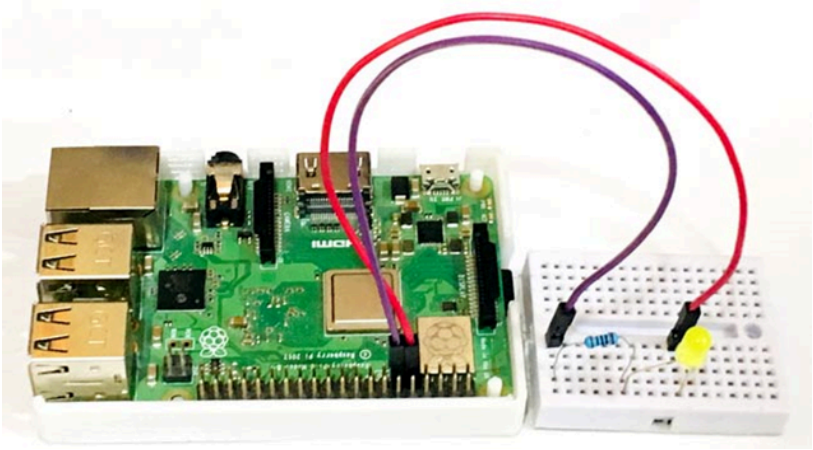
As mentioned during the introduction, our demo project for today will be to **control the Raspberry Pi's GPIO using a Node-RED flow**. To show the

variation in the GPIO's state, we will be connecting an LED to the GPIO such that when that particular GPIO pin is turned on, the LED comes on and vice versa.

Connect the LED to the Raspberry PI as shown in the schematics below.



I also build the same on my hardware using a breadboard, LED, resistor and some connecting wires. My hardware set-up looks something like this once the connections are made.

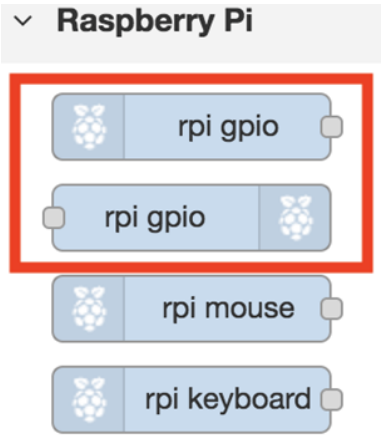


This project can be easily converted into a **Home automation project,** just by replacing the LED with a relay and any AC appliance, learn how to do that by going through various Home Automation Projects.

Creating a Flow in Node-RED

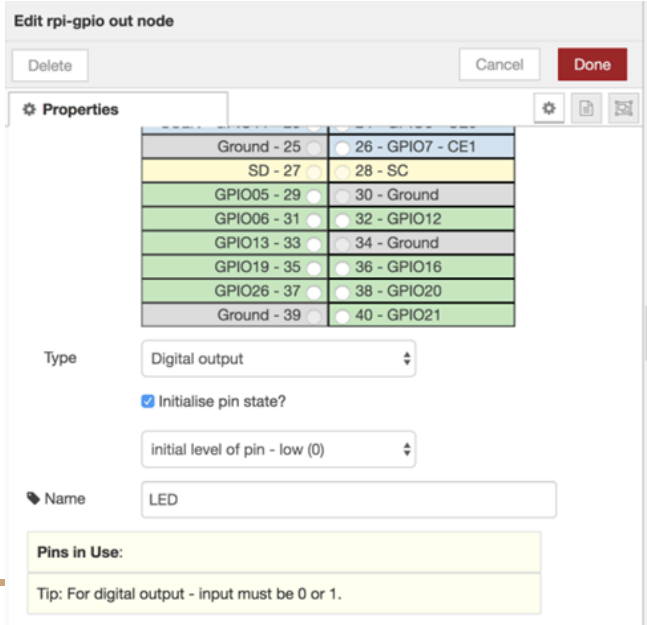
With the LED connected, we can proceed to develop the flow. **Programs in NodeRed are called flows** just like the Arduino IDE calls them sketches. Flows are created using a combination of nodes. You can create multiple flows which

can all run at the same time, but for this tutorial, we will be creating a single flow to turn the LED on/off.



To begin, Scroll to the bottom of the Nodes palette, you will see the raspberry pi nodes section towards the end with two nodes labeled **rpi gpio.** These nodes are used for communicating with the raspberry pi's GPIO. One of the Nodes is for input while the other is for output differentiated by the position of the raspberry pi's logo. For the **input node**, the logo comes before the text, while for the output **node** the logo comes after the text as shown in the image below.

For this tutorial we will be making use of the output node, drag it into the flow section of the interface. This step is similar to declaring a particular pin of your Arduino as Output using the *pinMode()* command. Double-click on the Output node and a pop up window will open as shown below to allow you edit the properties of the node.

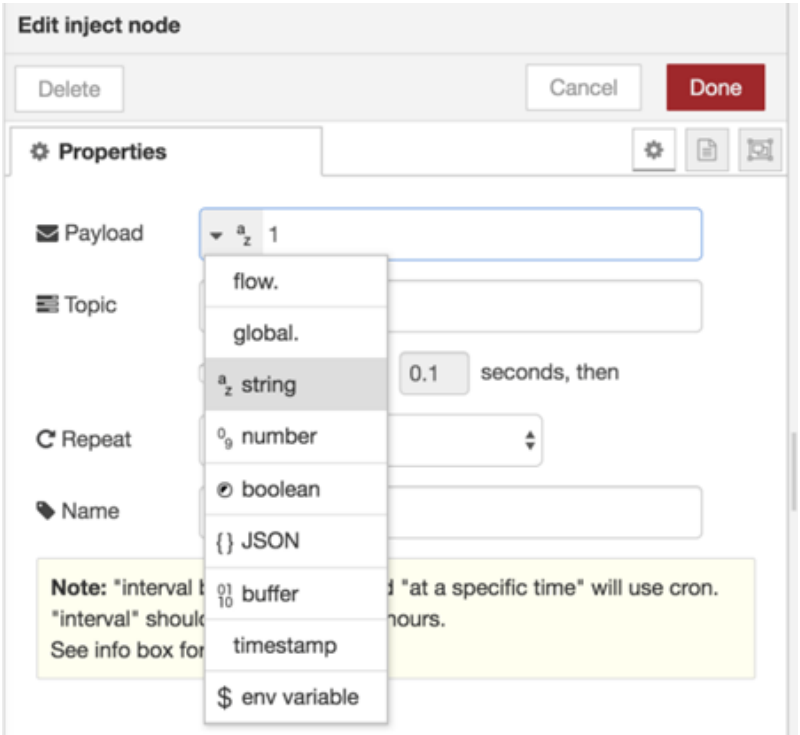


Under the pin property section, **select GPIO17 (pin 11).** Then set the type property to "**digital Output**" and tick the "**Initialize pin state?"** check box leaving the "**initial level of pin**" option as low (0). Give the node whatever name you like and click the done button.

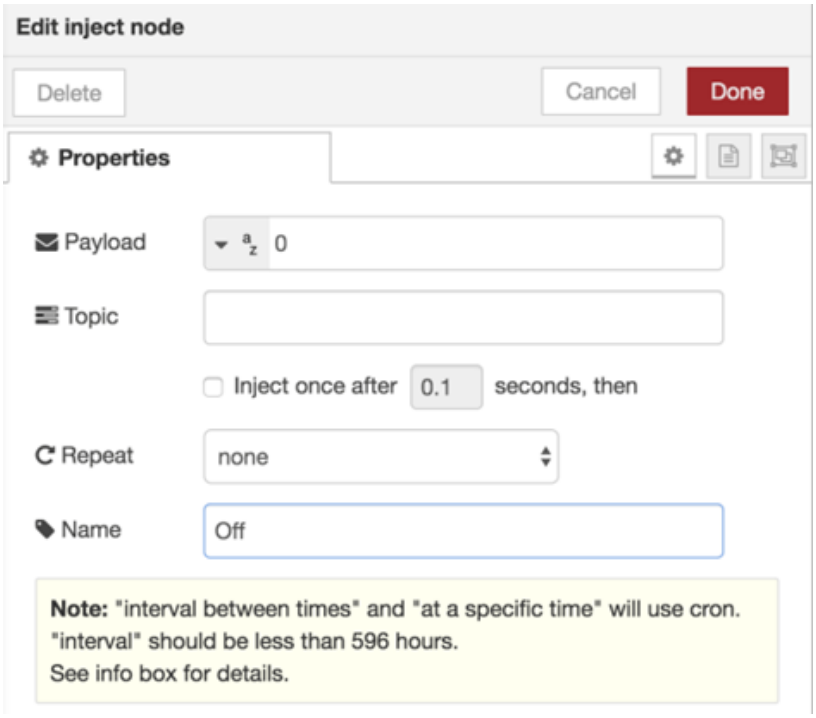
The name of the node should automatically change to the one you entered under the properties setting. For example I named it as LED and hence the node name is also changed as LED as shown below.

To turn the LED on/off, we need to use an input, something to drive the action. While we could use a push button, I want to use this to introduce the feature in Node-RED that allows the injection of messages into flows. This feature is called the **inject Node**. We will be using two inject nodes. One will be to turn the LED on while the other will be to turn it off.

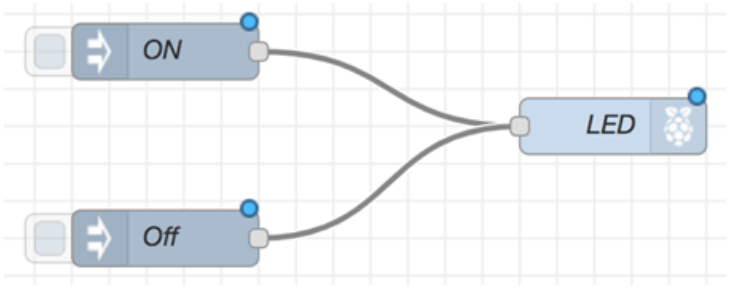
Go to the node palette and drag the inject node to the flow. It's the first node in the palette with an arrow, the inject node is highlighted in the image below. Double-click on it to edit its properties. Change the data type to string by clicking the dropdown in front of payload, and Enter 1 in the Payload box. The value in the payload box is what will be injected into the flow when the node is pressed. Set the name of the node to "ON" Press the "Done" button to save.



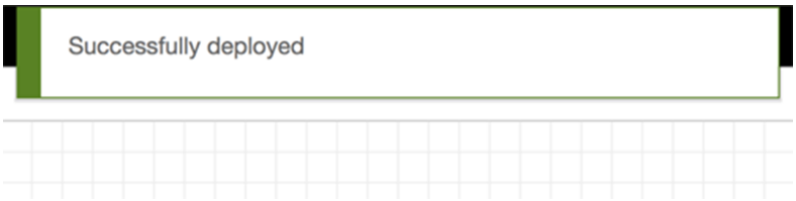
Repeat the above for the second node, setting the payload value to “0” and its name as “off” as shown below.



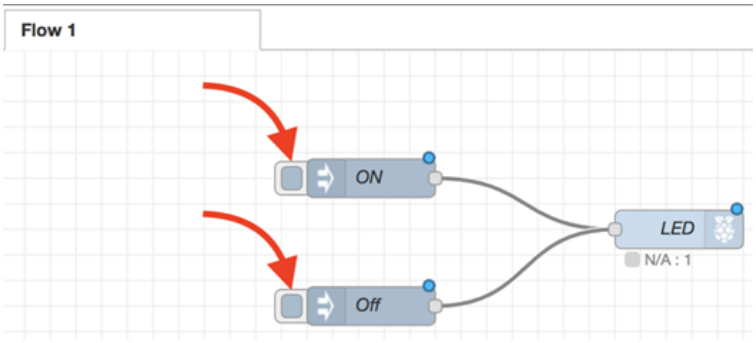
Under the properties function, **the *repeat* dropdown can be used to automate the injection** so the buttons are pressed at intervals. **This can be used to create the blink effect**. Join the nodes together as shown below, by dragging the grey dot on one of the nodes to the other to create the flow.



With that done, we have now completed our very first Node-Red Flow. Next step is for us to **deploy the flow on the raspberry pi**. Click the red deploy button. You should see “successfully deployed” flash at the top of the screen as shown below.



Click the gray button behind the inject nodes to activate each of the nodes.

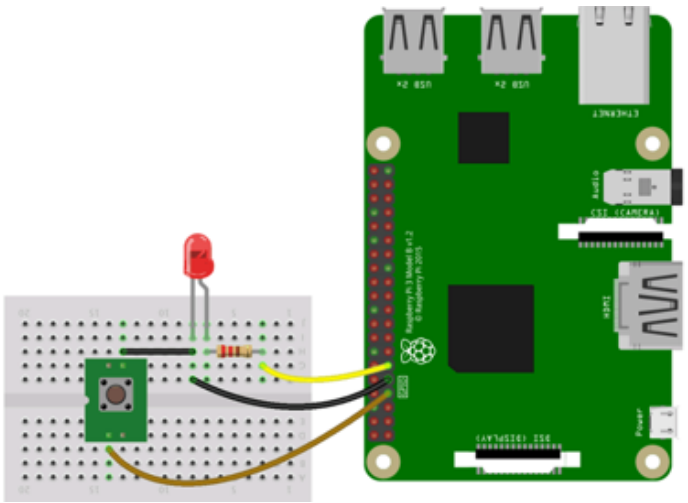


That's it. The complete working of this set-up can be found in the video linked at the bottom of this page.

Adding a Pushbutton/Switch in Node-RED interface with Raspberry Pi

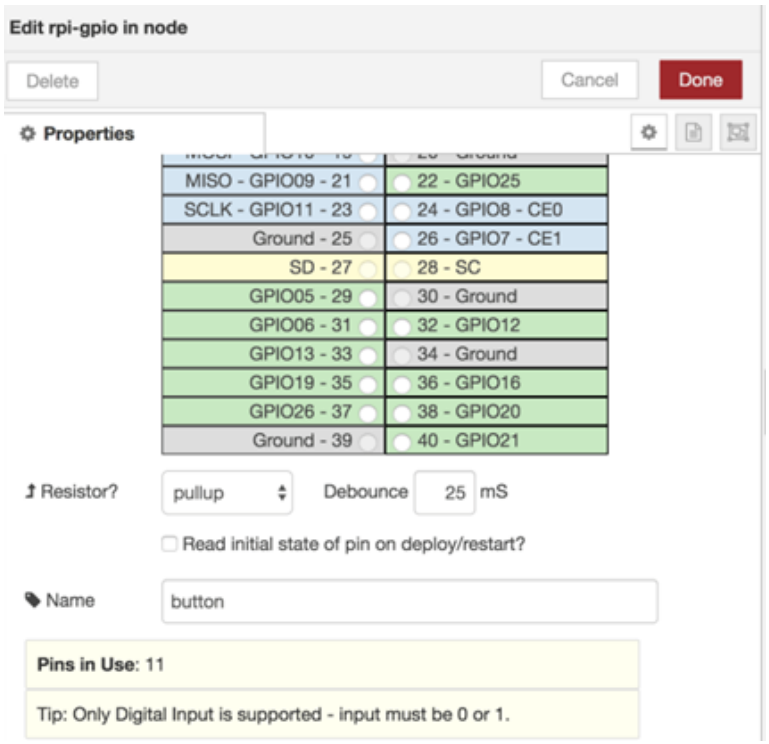
To explain how **to connect/add an input element to your flow**, we will be taking the above flow further by adding a pushbutton to replace the inject nodes.

Connect the pushbutton to the raspberry pi, such that one leg of the pushbutton is connected to ground and the other is connected to GPIO pin 4(pin 11) of the raspberry pi, as shown in the schematics below.



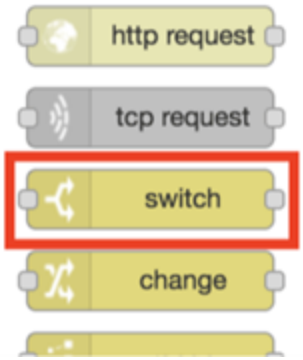
With this done, Go back to Node-Red and delete the two injected nodes we used earlier, by clicking on the nodes and pressing delete on your keyboard or double-clicking the nodes and pressing delete on the pop-up window. With that done, scroll down the node palette to the raspberry pi section and select the input node. It is the one with the raspberry pi icon on the left, before the node's name.

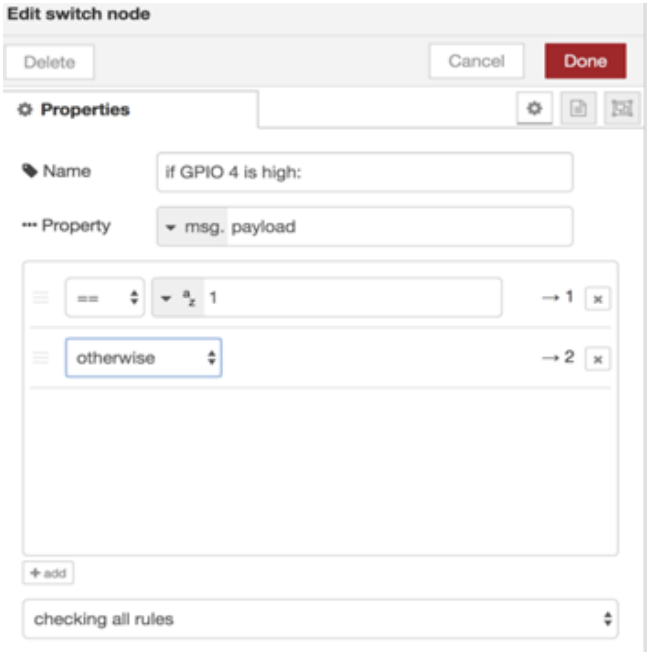
Drag the node into the flow and double click on it to edit properties. Set the pin to GPIO 4 (pin 11) and set the dropdown in front of the resistor to pull-up. This will "pull up" GPIO 4 to HIGH. Click on the done button after setting the properties.



To allow us to make **logical decisions** when the switch is pressed rather than just shorting GPIO4 to ground, we will be using the switch node. Search for it in the node palette, under the function section and drag to the flow.

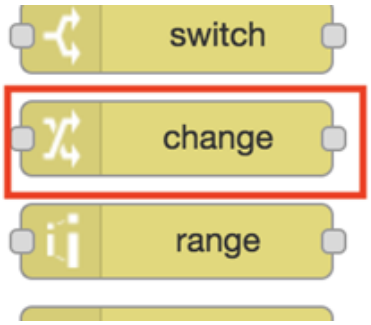
The switch node allows you to make decisions in a way similar to the “if” statements. It can be set up to have different outputs depending on the input value. For this tutorial, we will be configuring it with two output path, such that, when the msg.payload property is equal to 1 (switch is not pressed) it should follow the first path and a second path will be followed if any other input other than 1 is observed at the input(switch is pressed). Paths are added using the “+add” button. Double click on the node and configure as described above. Click done when finished.



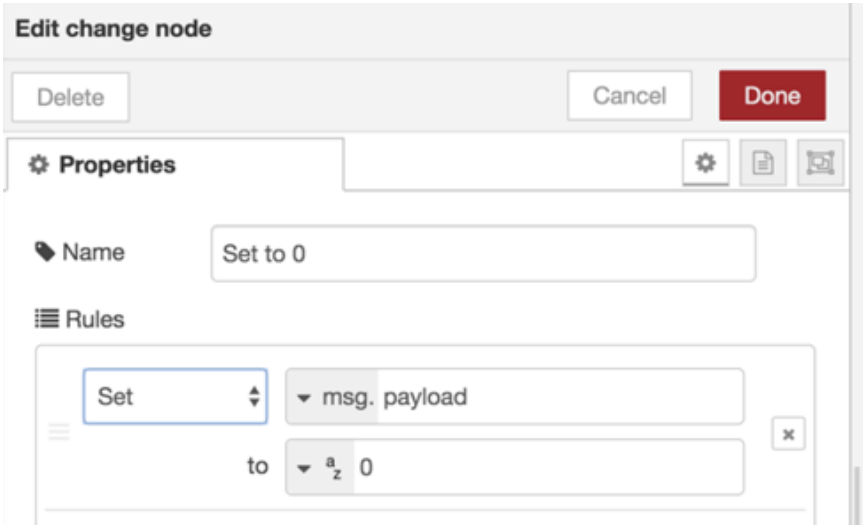


As soon as you hit the done button, you should see the two paths reflect in the outlook of the **switch node**, as it will now have two junctions at the output.

Next, we need to bring in a “change” node. The change node will be used to set the state of the LED based on the result of the argument **node**.

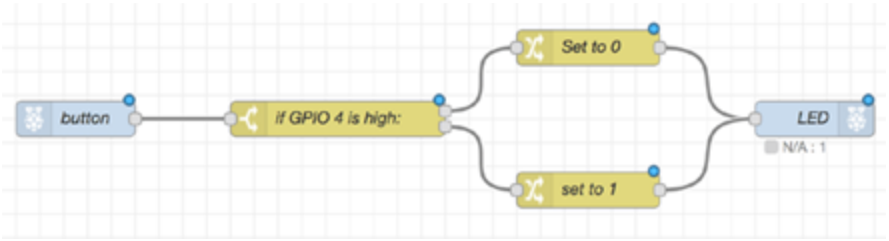


We will be using two of the change nodes. The payload of the first one will be set to 0 as shown in the image below and the payload of the second one set to one. Click done after editing properties for both.



To better explain, remember we used the pull-up resistor? Thus, when the button has not been pressed, the output at that pin will be HIGH(1), which means if we pass it through the switch node, The LED will be “on”, since this should not be so, we will use the “change” node to set it as LOW(0). The second “change” node is used to set the value to one such that when anything other than the HIGH state of GPIO pin 4 is detected, it should turn the LED “on” as this will mean the pushbutton has been pressed. Connect the change nodes and the rest of the nodes together as shown in the image below.

With this done, we are now ready to deploy the project. Go over the connection to ensure everything is as it should be, then click the deploy button. As usual, if



successful you should see the deploy successful popup and now be able to control the led using the switch.

While the Node-Red makes it easy and fast to prototype your build without worrying about the code, it may not actually be the best fit, especially for experienced developers who will want flexibility and control over the program. Nevertheless, it's a great tool that allows you to prototype builds in minutes.