Magicbit-Magicblocks.io Documentation

Release latest

Sep 06, 2020

Contents

The Internet of things is nothing new. It is an evolutionary next step in the machine to machine communication paradigm. With an ever increasing number of 'things' at our disposal the need for connected 'things' has grown rapidly. However with increasing options comes added complexity which introduces steep learning curves and more importantly more confusion. What devices should I use? What cloud platform will best server my interest? I am new to IoT where should I start? Those are the question that would linger in any one who would dare to wonder in to the internet of Things. We, at magicblocks.io have put some magic to work to make the life of both the newcomer and the professional one step easier.

CHAPTER 1

Magicblocks.io Introduction

Magicblocks.io is the IoT platform by A&T Labs for building IoT applications with minimal hassle. Tried and tested over the years the lateset release of magicblocks.io will connect to more devices and do more thanks to the open source project Node-Red. magicblocks.io is a launchpad for learning and prototyping your internet of things. It consists of

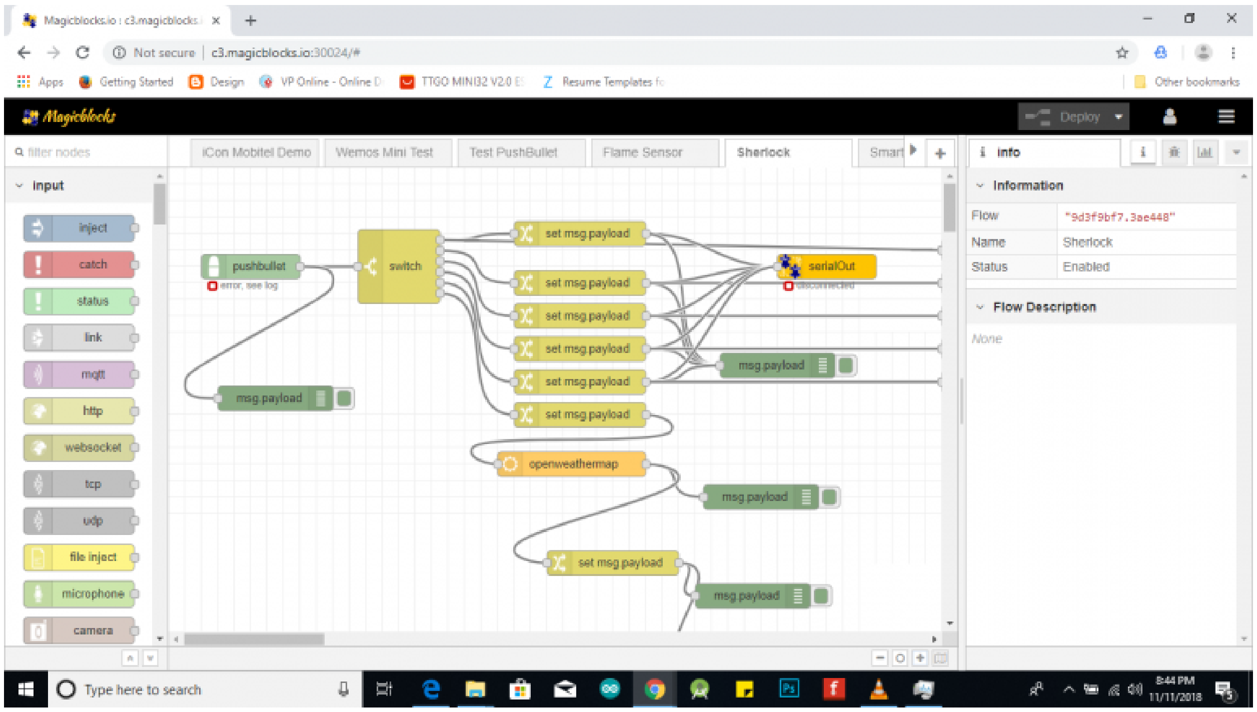
- **Hardware suite made up of**
  - Development boards
  - Prototyping kits for sensors & actuators
- **Cloud platform made up of**
  - Drag and drop editor to easily cook up your solutions
  - Dashboards to visualize your data
  - Data storage
  - API services

Everything has been designed to make the learning curve as shallow as possible for the newcomer and as flexible as possible for the advanced user.

1.1 Magicblocks.io Playground

Any ioT creation you want to connect with nodes without any coding and your project running in the cloud platform the development board you use will not have much memory or more processing capability, unless you want to create Image Processing, Character Recognizing, DB Handling, you can create any project on it.

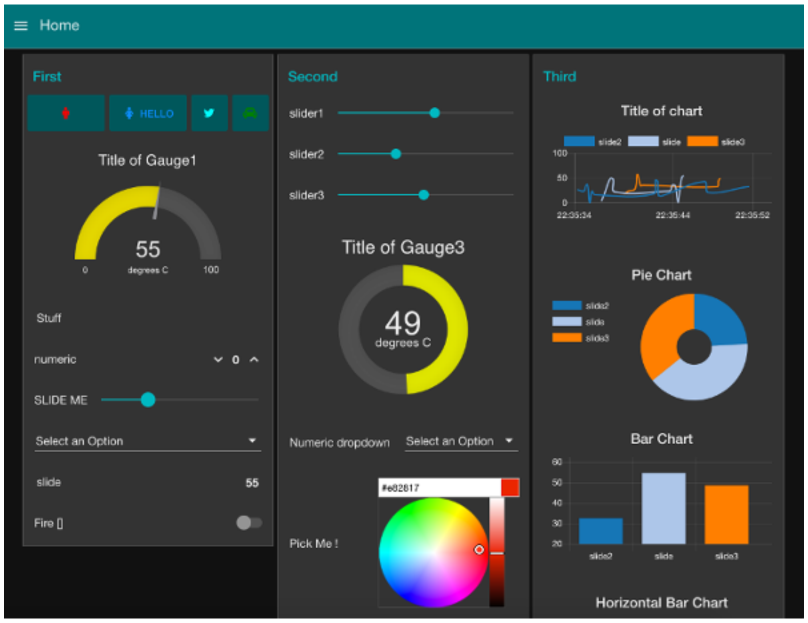
You can also create attractively online dashboards online without any coding



1.2 Magicblocks.io Dashboard

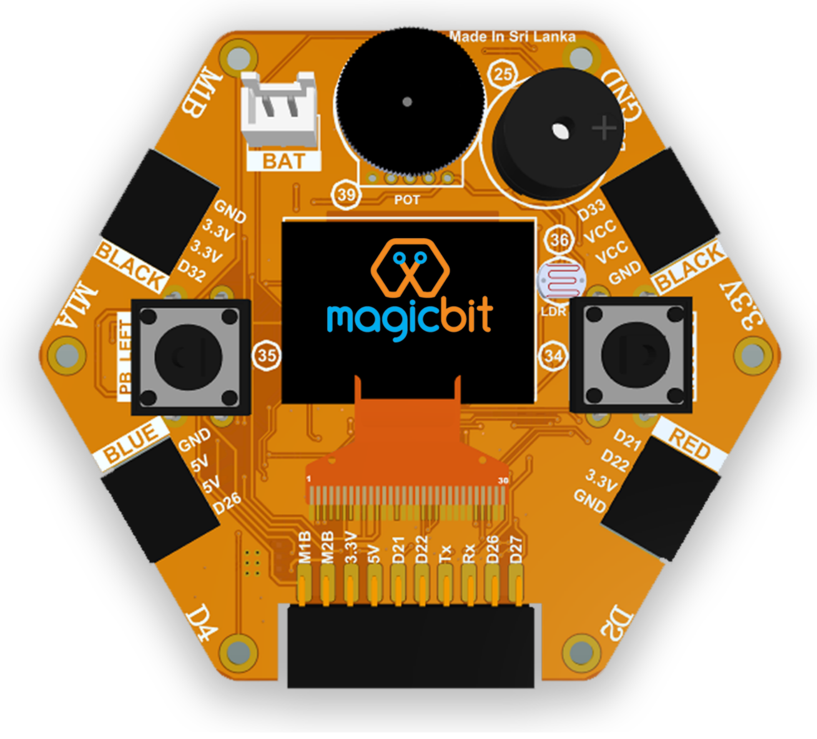
In Magicblocks you can also create Web based-dashboards attractively for your IoT designs without any codes. You need to connect the necessary widgets to the dashboard that you want and to configure the settings you need to reach them. Then, you can access your Online Dashboard from the dashboard menu on your dashboard, where you can click on the URL in the browser to access the Dashboard via any Device Device such as Smart Phone, Tablet, PC. Any ioT creation you want to connect with nodes without any coding and your project running in the cloud platform the development board you use will not have much memory or more processing capability, unless you want to create Image Processing, Character Recognizing, DB Handling, you can create any project on it.

You can also create attractively online dashboards online without any coding


Home
First
HELLO
Title of Gauge1
55
degrees C
0
100
Stuff
numeric
0
SLIDE ME
Select an Option
slide
55
Fire
Second
slider1
slider2
slider3
Title of Gauge3
49
degrees C
Numeric dropdown
Select an Option
#e82817
Pick Me !
Third
Title of chart
slide2
slide
slide3
Pie Chart
Bar Chart
Horizontal Bar Chart

CHAPTER 2

What is Magicbit



This device is an Integrated development platform for learning and solution designing of electronics, robotics, Internet of Things and coding. The device can interact with a host of sensors such as light sensors, fire sensors, motion detectors etc. and output devices such as LEDs, switches, buzzers, speakers, motors etc. The hardware can be programmed using industry standard integrated development environment (IDE) that runs on PCs, Tablets, Mobile devices, or web platforms. This device supports a large number of programming platforms such as C++, python, scratch, magicblocks, mblock and codda, Hence the learning curve to learn to operate and utilize this device is shorter. Ardunio & MicroPython are for users with prior programming experience. Magicbit provides extension for mBlock3 for kids without programming knowledge. Codda is a visual programming language which same time can experience the true coding. Magicblocks is a NodeRed based platform for IoT solution design for any user group.

This device also has the following special features:

- In-built battery charger, WiFi & Bluetooth connectivity;
- Integrated sensors and actuators to enable users to test and design projects without additional components;
- An internal OLED display;
- Plug & play feature to easily connect accessories;
- An enclosure for productization of designs

Brain of the Magicbit is ESP32, which is a series of low-cost, low-power system on a chip microcontrollers with integrated Wi-Fi and dual-mode Bluetooth. Therefore any project or document available on internet which supports ESP32 is supported for Magicbit as well.

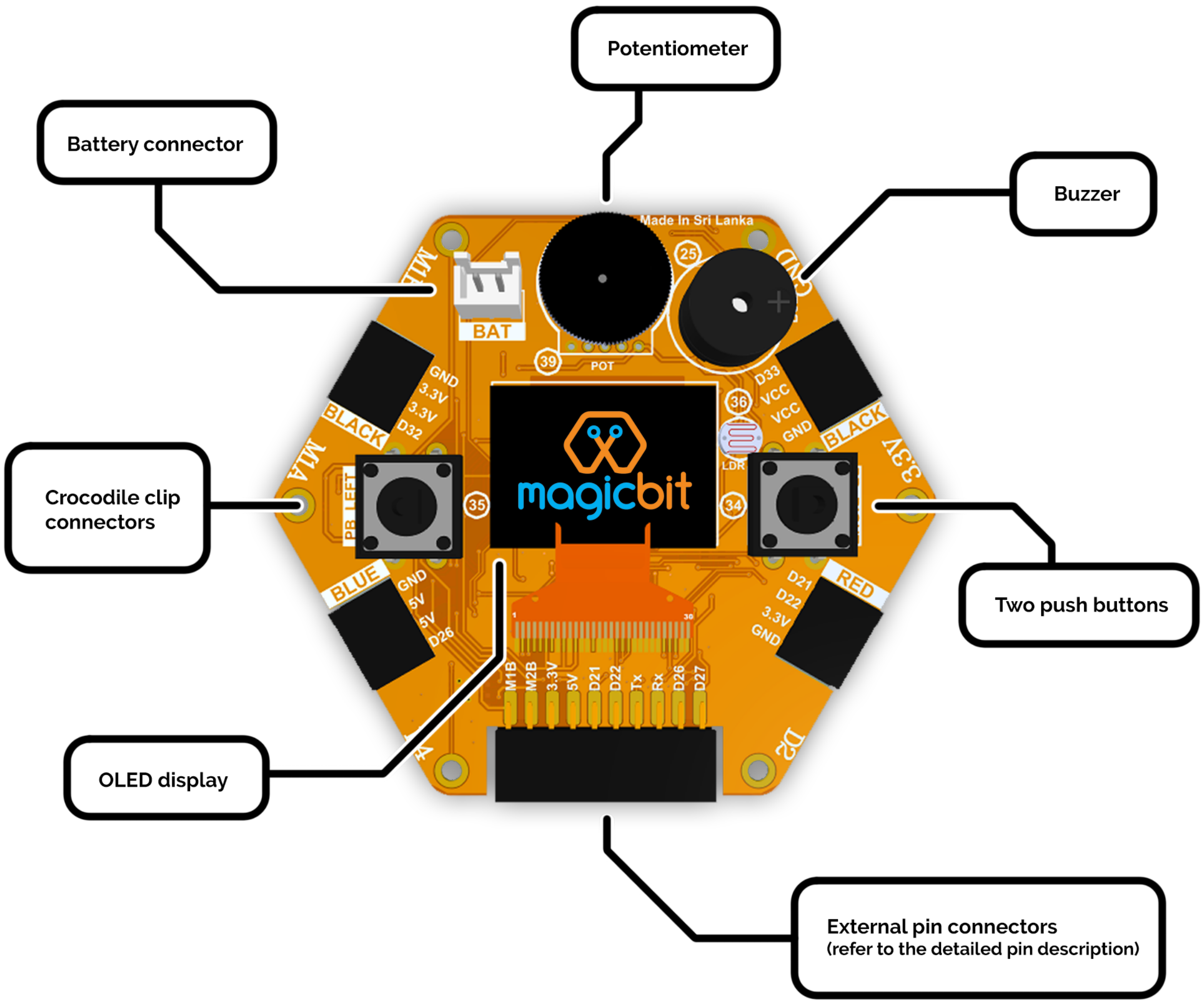
CHAPTER 3

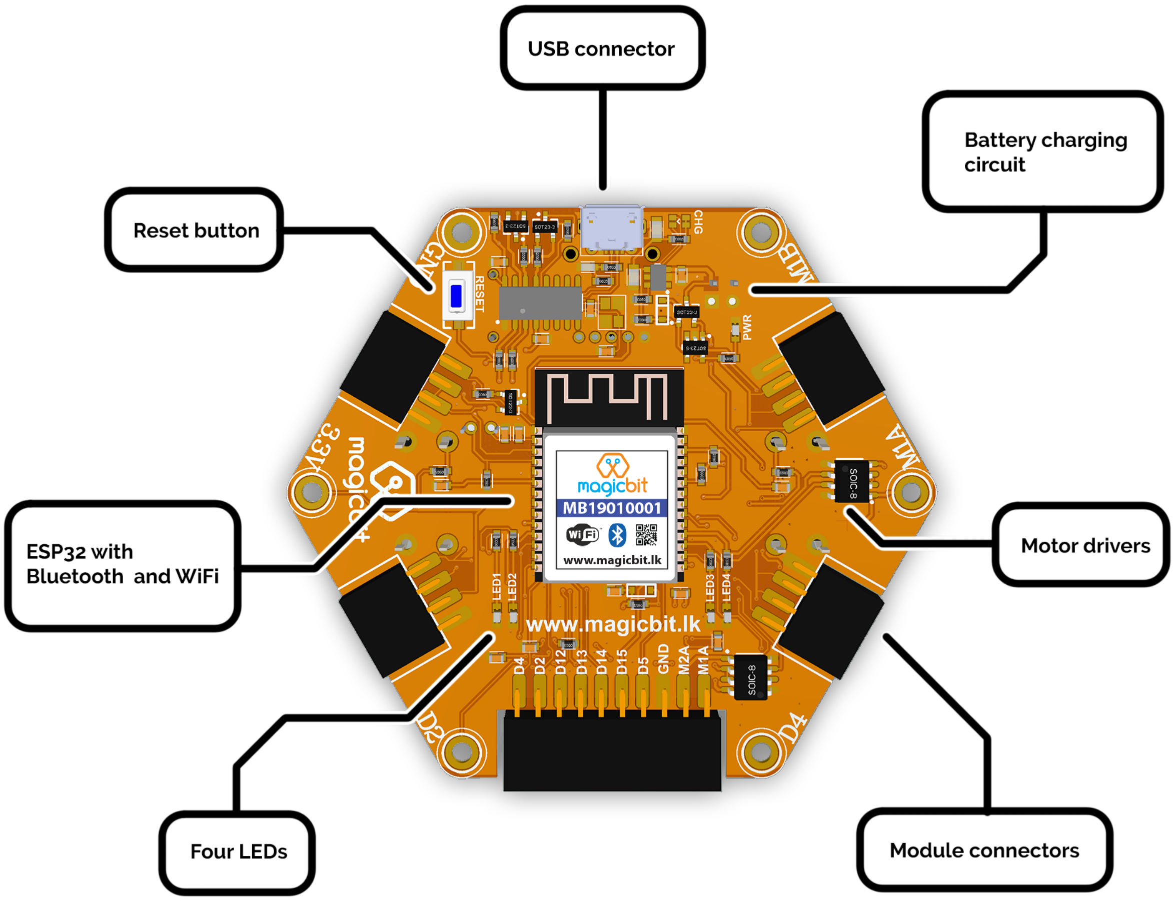
Hardware

3.1 Specifications

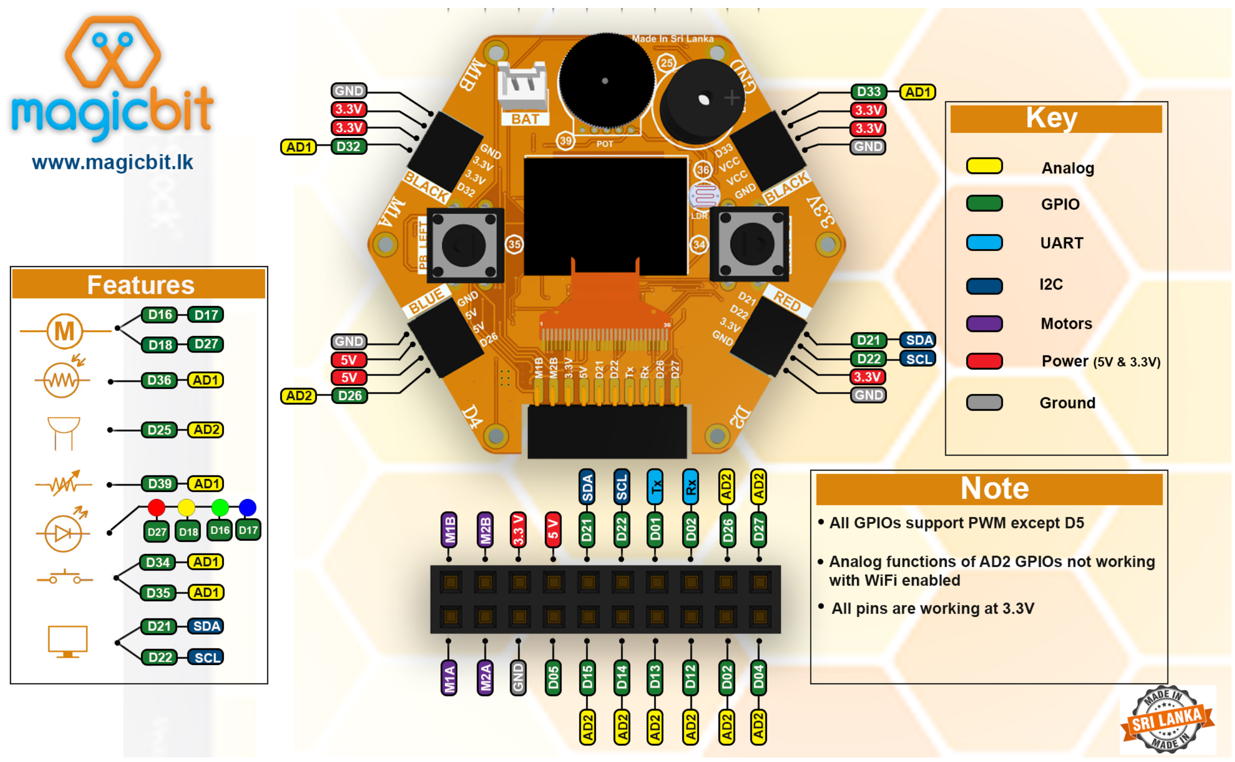
- **Processor** - Xtensa dual-core
- **Speed**- Up to 240Mhz
- **Flash Memory**-4MB
- **Ram**-520KB
- **Inputs**-Pushbutton, LDR, Potentiometer
- **Outputs**-LEDs, OLED Display, Buzzer
- **Other**- Dual Motor Driver, Li-Ion Charger
- **Connectivity**- USB, WiFi, Bluetooth

3.2 Layout


Potentiometer
Battery connector
Buzzer
Crocodile clip connectors
Two push buttons
OLED display
External pin connectors
(refer to the detailed pin description)
magicbit
BAT
POT
LDR
Made In Sri Lanka
M1B
M2B
3.3V
5V
D21
D22
Tx
Rx
D26
D27


USB connector
Battery charging circuit
Reset button
ESP32 with Bluetooth and WiFi
Motor drivers
Four LEDs
Module connectors
www.magicbit.lk
RESET
PWR
CHG
LED1
LED2
LED3
LED4
D4
D2
D12
D13
D14
D15
D5
GND
M2A
M1A
M1B
3.3V
magicbit
MB19010001
SOIC-8

3.3 Pinmap



Note

- All GPIOs support PWM except D5
- Analog functions of AD2 GPIOs not working with WiFi enabled
- All pins are working at 3.3V

3.4 Features

3.4.1 LED

There are four leds on backside of the Magicbit with color red, yellow, green & blue. A LED(light-emitting diode) is a semiconductor light source that emits light when current flows through it. Blinking a LED is the hello world to the microcontroller programming world.

3.4.2 BUTTON

There are two buttons on the front of the Magicbit. The push-button is a component that connects two points in a circuit when you press it. The example turns on an LED when you press the button.

3.4.3 LDR

There is a LDR on the front of the Magicbit. LDR(Light Dependent Resistor) is a light-controlled variable resistor. The resistance of a photo-resistor decreases with increasing incident light intensity. You can measure light intensity using LDR as a analog output.

3.4.4 POTENTIOMETER

The potentiometer is a component with rotating contact that forms an adjustable voltage divider. A potentiometer is a simple knob that provides a variable resistance, which we can read into the Magicbit board as an analog value.

3.4.5 DISPLAY

OLED (Organic Light Emitting Diodes) is a flat light emitting technology. OLED display has a film of organic compound that emits light in response to an electric current.You can display varoius graphics and text on the display.

3.4.6 BUZZER

There is a buzzer on the front of the Magicbit. Buzzer is an electronic device commonly used to produce sound.

3.4.7 BATTERY

There is a Battery connector on the front of the Magicbit.Single cell rechargeable li-ion battery (3.7V) can be plugged in to a battery connector to puwer the Magicbit. Battery can be recharged by providing USb power to the Magicbit.

3.4.8 MODULES

There are four module connectors on the edge connector of the Magicbit, which we refer to as ports. Which can connect various accessories to Magicbit board and program to work with Magicbit. Matching accessory pin connector color marked on the Magicbit. As an example module with blue pin connector should plug in to blue port of the Magicbit.

3.4.9 USB

There is a micro USB port on the back of the Magicbit.Connect the micro USB port to a mobile phone charger or computer through a cable and it will draw power required for the board to function and it also used program magibit and data transferring with a computer.

3.4.10 WiFi

WiFi is a technology that uses radio waves to provide network connectivity. Magicbit consists with wifi module. WiFi technology has widely spread lately and you can get connected almost anywhere; at home, at work, in libraries, schools, airports, hotels and even in some restaurants enabling IOT connectivity capabilities.

3.4.11 BLUETOOTH

Bluetooth is a wireless technology standard used for exchanging data between fixed and mobile devices over short distances using short-wavelength UHF radio waves. Magicbit consists with wifi module which enables IOT connectivity capabilities

3.4.12 EXPANSION HEADER

Magicbit can connect various electronic sensors, electronically controlled actuators,etc to Magicbit via these external connectors

3.4.13 CROCODILE CLIP

Magicbit crocodile clip connectors used to connect an electrical cable to a battery or some other component. Functioning much like a spring-loaded clothespin, the clip's tapered, serrated jaws are forced together by a spring to grip an object

3.4.14 RESET BUTTON

In electronics and technology, a reset button is a button that can reset a device. On Magicbit, the reset button restarts the Magicbit's programme

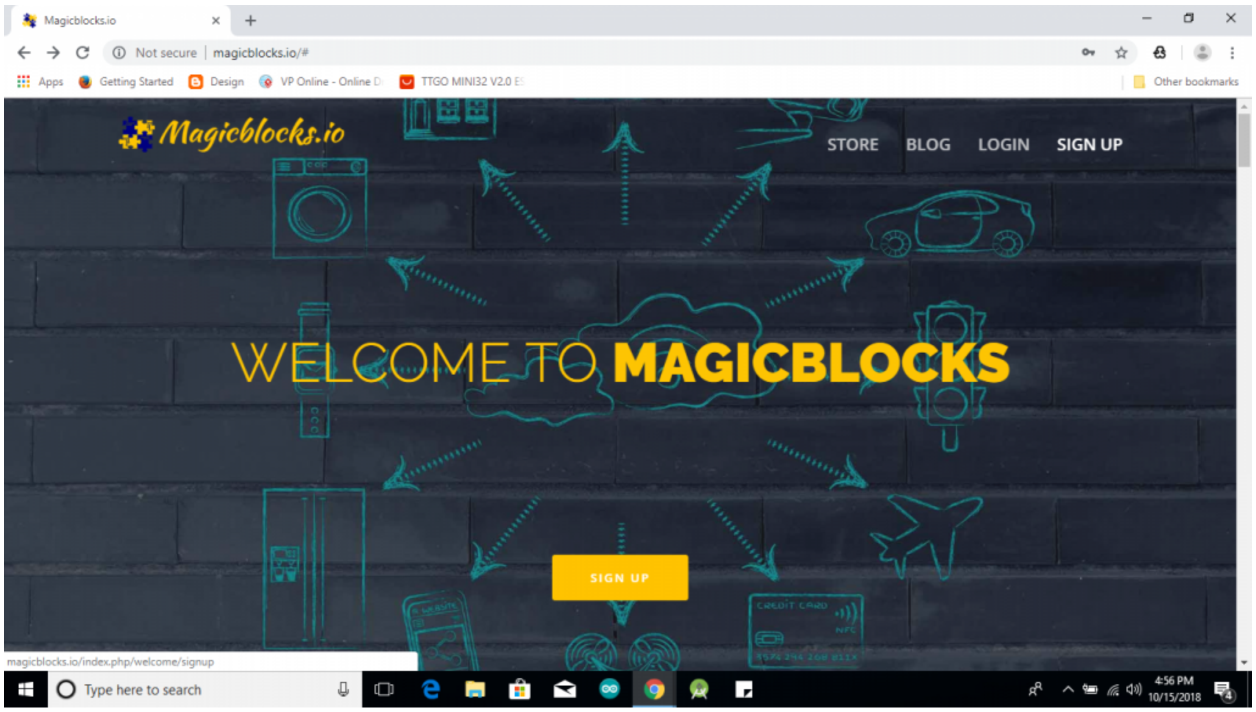
CHAPTER 4

Getting Started

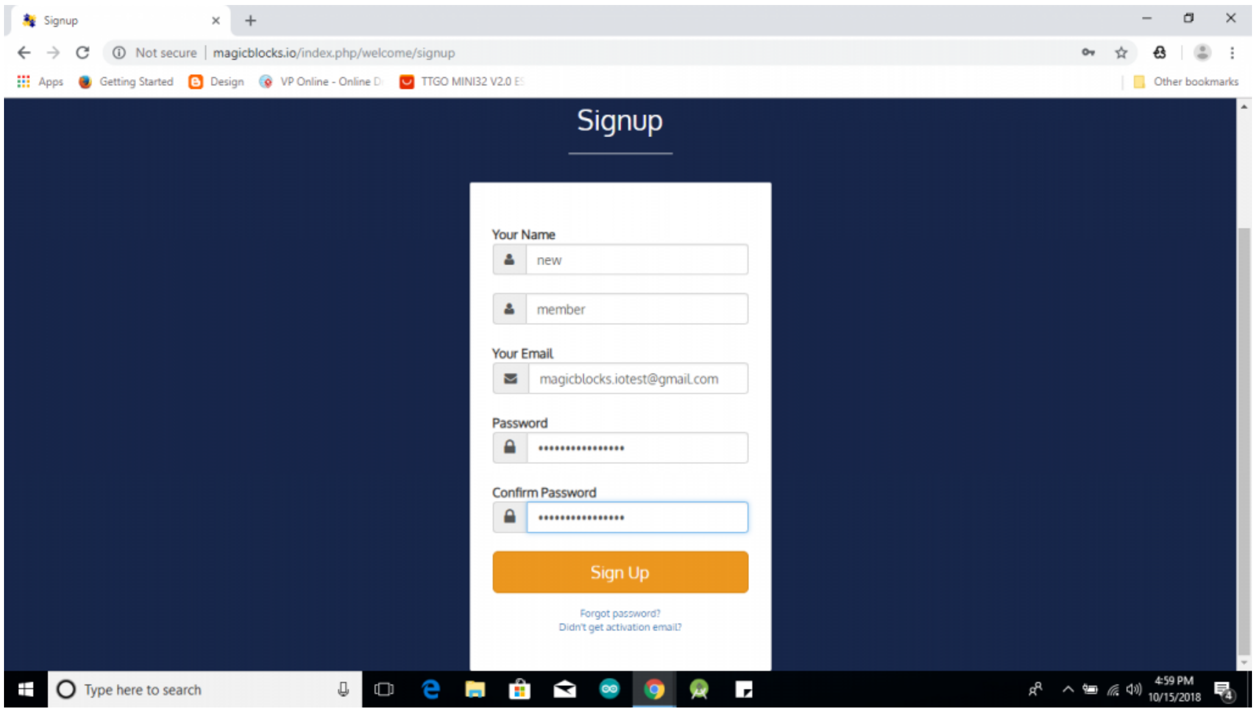
Watch Video

4.1 Create Magicblocks.io Account

- Go to magicblocks website http://magicblocks.io/
- Select SIGNUP


Magicblocks.io
Not secure | magicblocks.io/#
STORE
BLOG
LOGIN
SIGN UP
WELCOME TO MAGICBLOCKS
SIGN UP
magicblocks.io/index.php/welcome/signup

- Click on the SIGN UP button after inserted your details


Signup

Your Name

new

member

Your Email

magicblocks.iotest@gmail.com

Password

••••••••••••••••

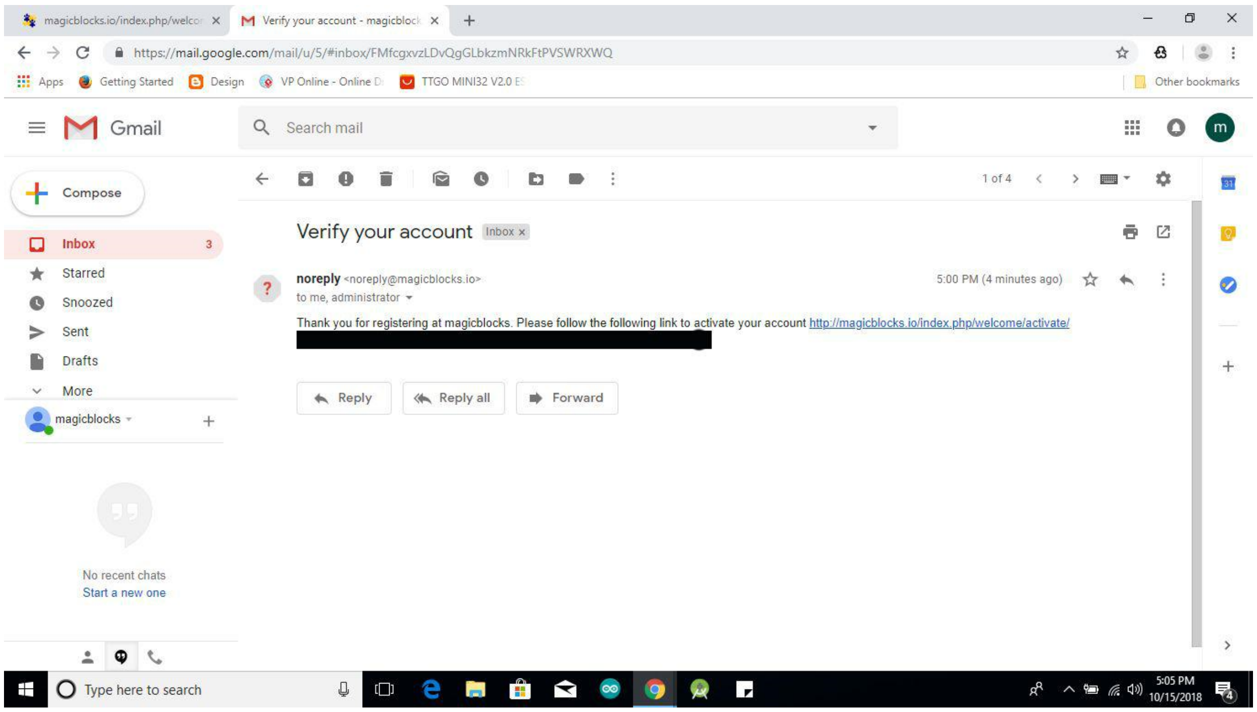
Confirm Password

••••••••••••••••

Sign Up

Forgot password?
Didn't get activation email?

- Go to the email account you provided and activate your Magicblocks account with the Activation Link
- Follow the Activation Link which we have sent you as an email.



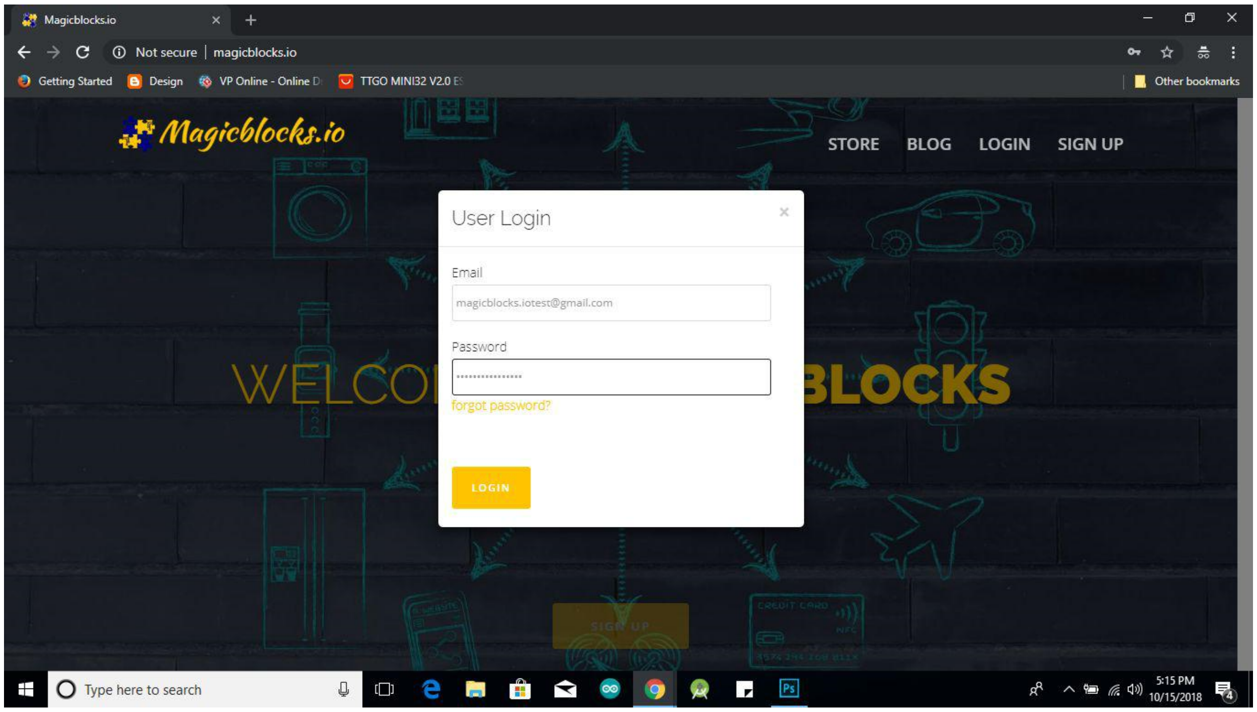
Verify your account Inbox

noreply <noreply@magicblocks.io>
to me, administrator
5:00 PM (4 minutes ago)

Thank you for registering at magicblocks. Please follow the following link to activate your account http://magicblocks.io/index.php/welcome/activate/
[redacted]

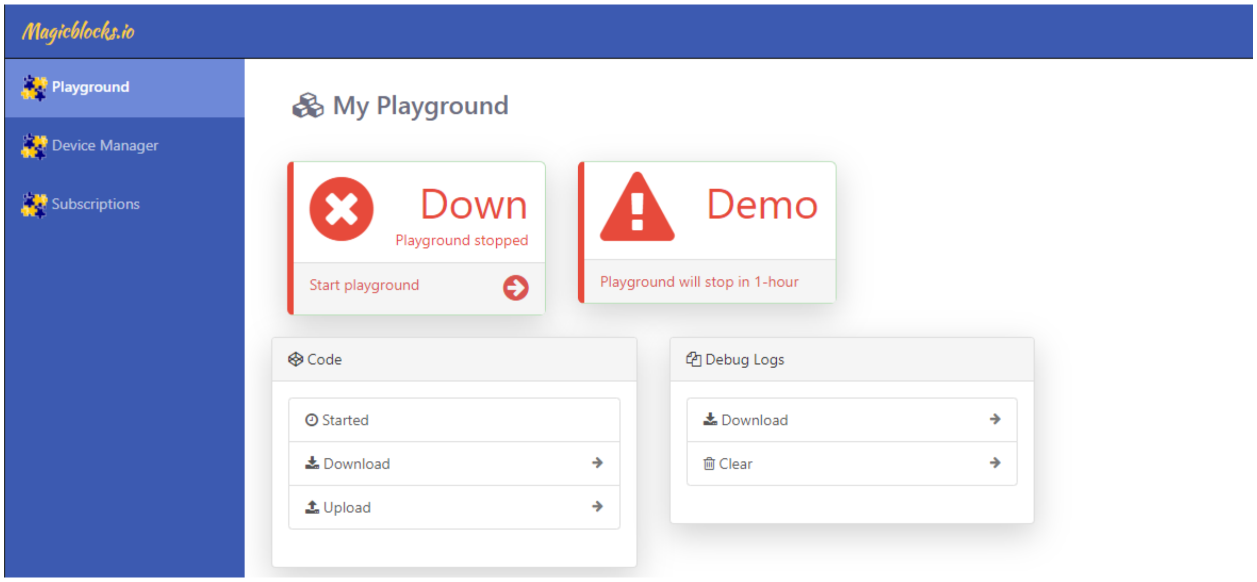
4.2 Login to Your Magicblocks.io Account

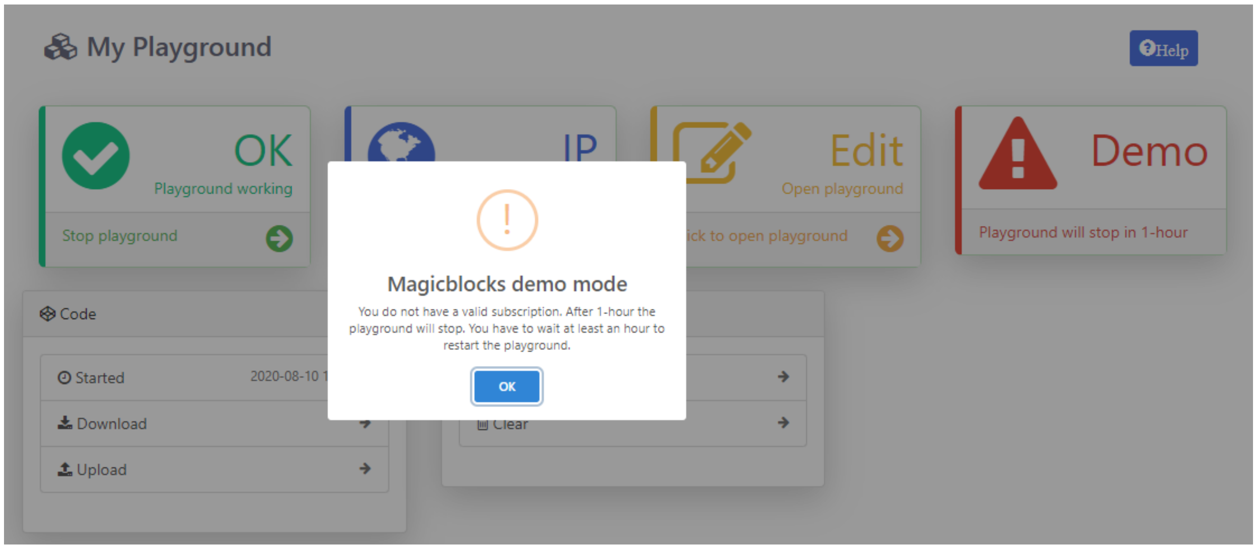
- Go to Magicblocks.io official website. www.magicblocks.io
- Select LOGIN
- Provide your login details.
- Enter your email address and the Magicblocks Password and sign in to magicblocks



4.3 Start the Playground

When you login for the first time your playground will not be running. Playground is the visual programming environment based on Node-Red that has been customized for seamless integration with hardware devices to enable IoT. If you do not have a valid subscription, you will be allowed to run the playground only for 1 hour continuously before it is automatically stopped. You will need to restart the playground manually after this 1 hour period. You can subcribe to 3 months free subscription by enterting coupon code in Subscription tab provided with your magicbit device. If you have any issue please write to info@magicbit.cc



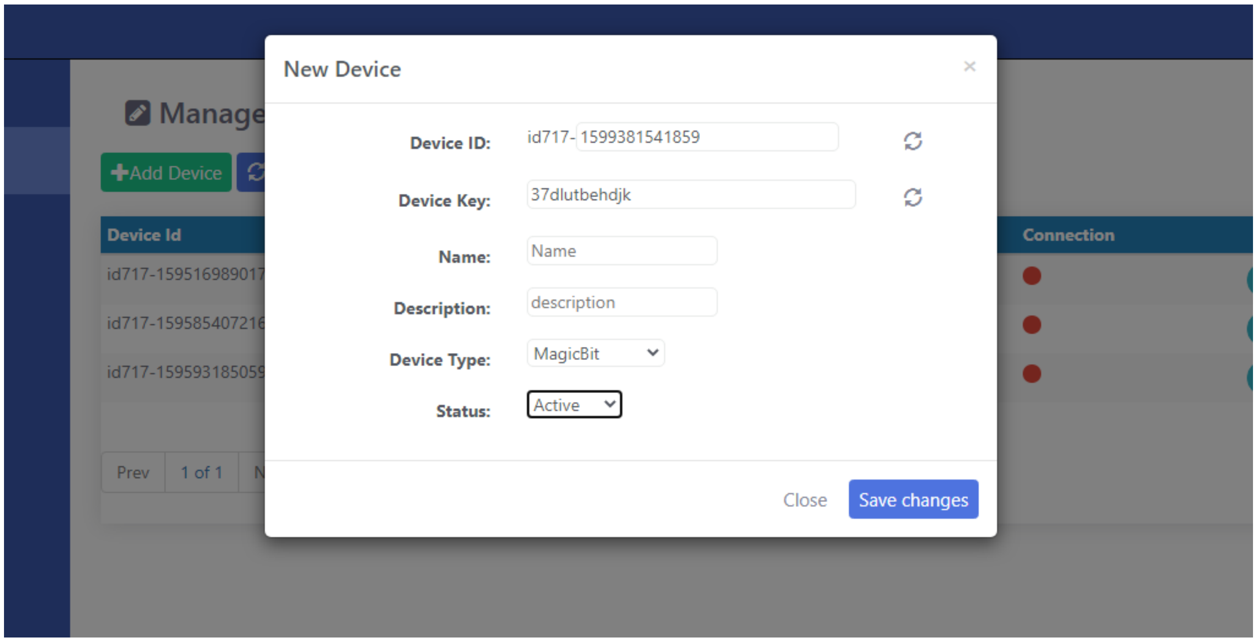


Magicblocks demo mode

You do not have a valid subscription. After 1-hour the playground will stop. You have to wait at least an hour to restart the playground.

4.4 Create a new device

Go to Device Manager and add a new device.. Set device type as **Magicbit** & set status to **Active** . You can use any name and description.



New Device

Device ID: id717- 1599381541859

Device Key: 37dlutbehdjk

Name: Name

Description: description

Device Type: MagicBit

Status: Active

Keep this browser tab open since you will need to copy the device Id and the key to setup the device in the next step.

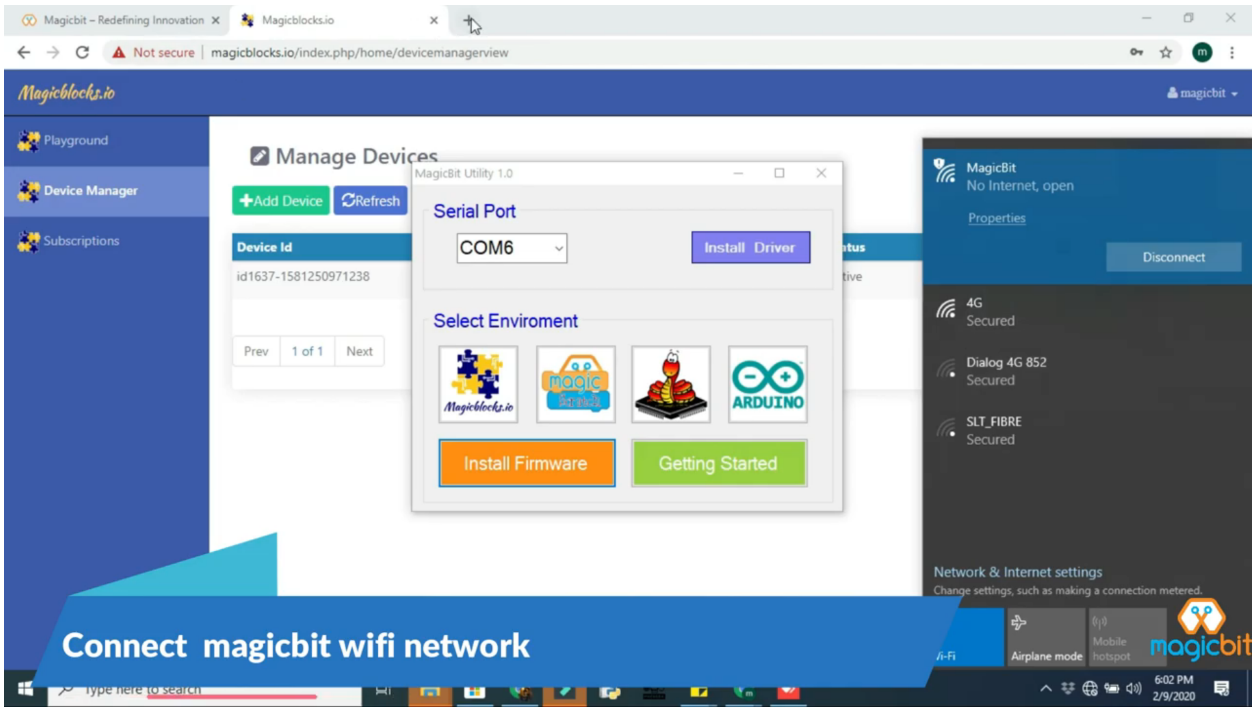
4.5 Setting up a device

Connect Magicbit to computer using provided USB cable.

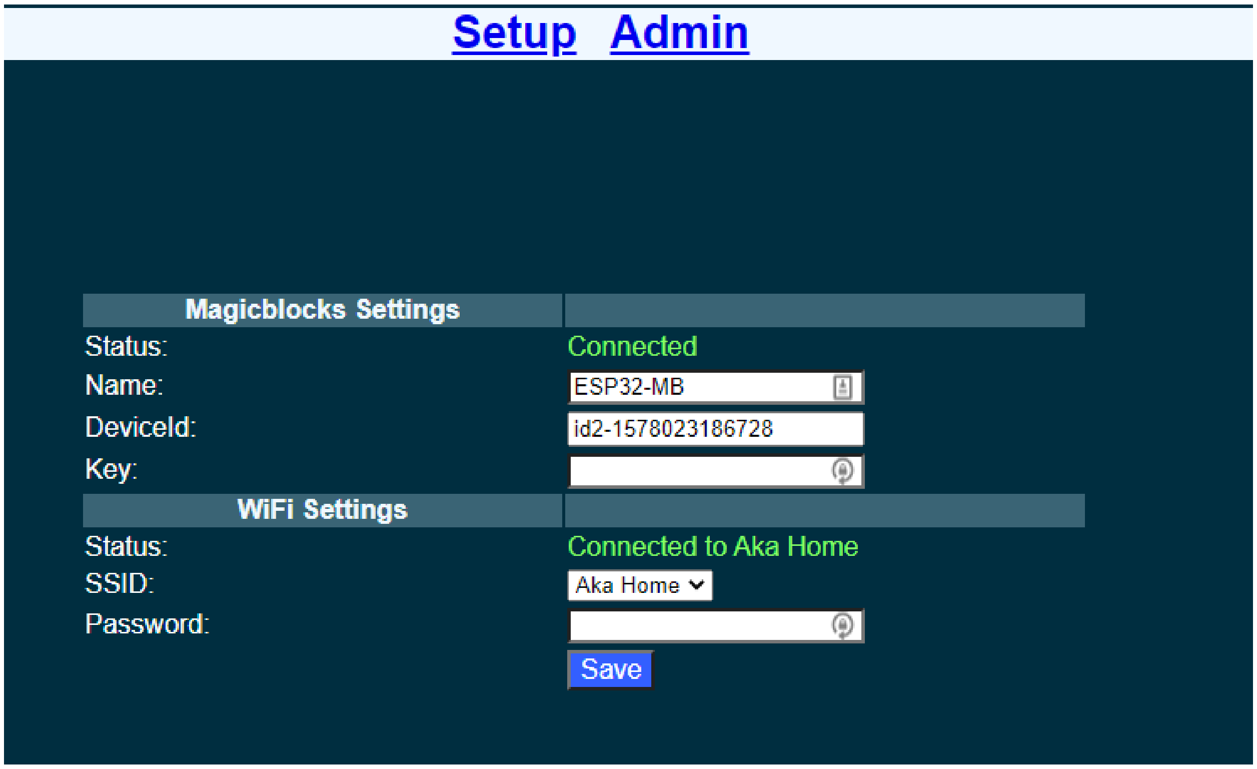
Download Magicbit utility tool and install drivers

Select COM port from drop down list and select magicblocks to install firmware. .. image:: Images/Utility.PNG

- Wait until uploading is completed and search for its configuration WiFi access point to come up. It would be named "Magicbit" in the default configuration



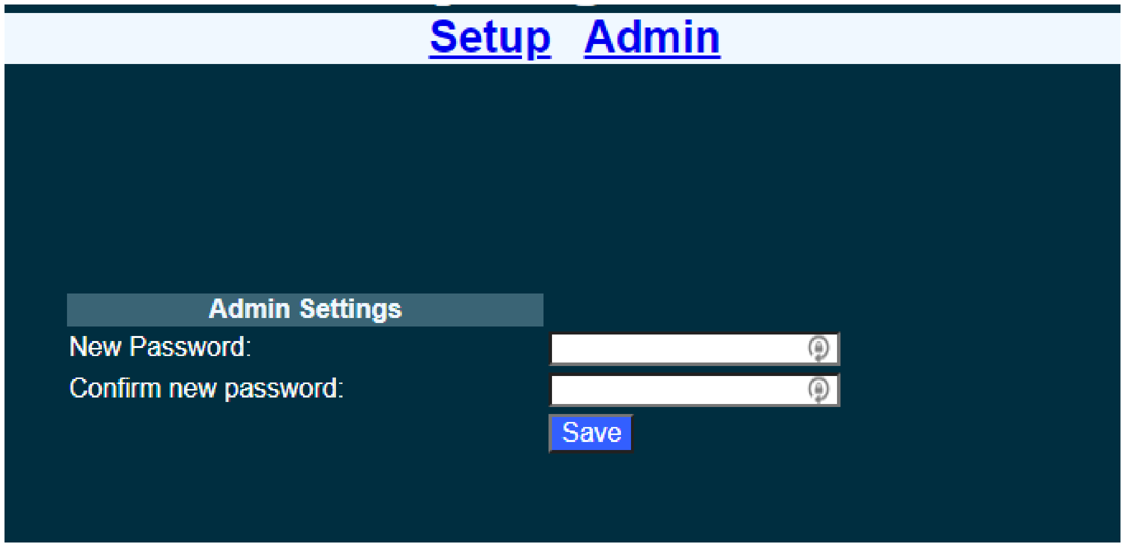
- Connect to this WiFi access point and go to http://192.168.4.1



Setup Admin

Magicblocks Settings	
Status:	Connected
Name:	ESP32-MB
DeviceId:	id2-1578023186728
Key:	
WiFi Settings	
Status:	Connected to Aka Home
SSID:	Aka Home
Password:	
	Save

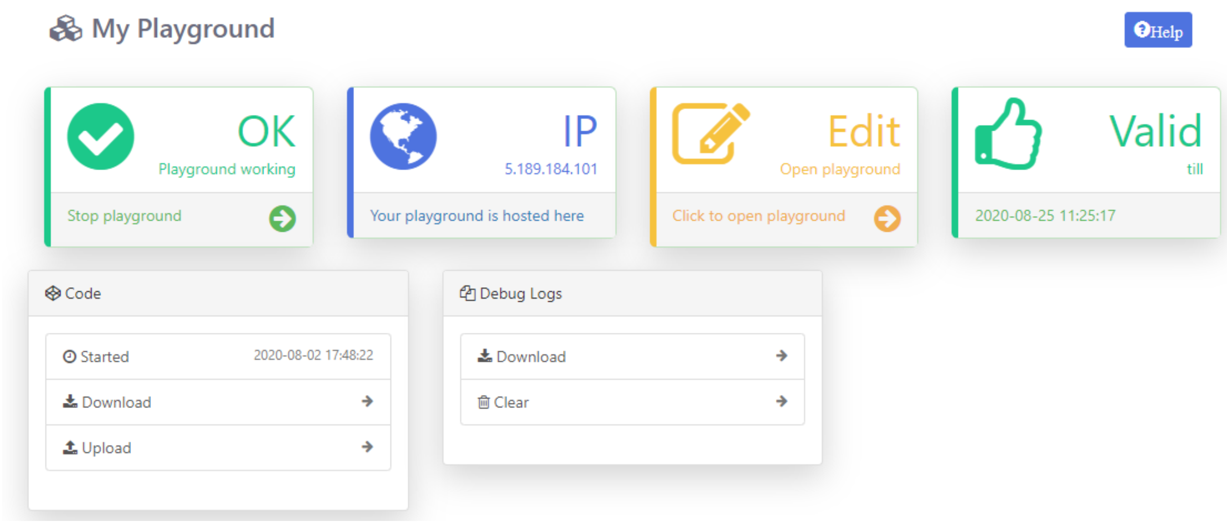
- Fill in the device Id and the key that was generated in the previous step. You can change the **Name** to any name you like up to 32 characters. Select the WiFi SSID from the drop down and enter the WiFi password. If your WiFi is not listed here make sure the WiFi AP is active and reset the Magicbit board and retry.
- Save the configuration. The default admin password is 12345. You can change this from the **Admin** tab



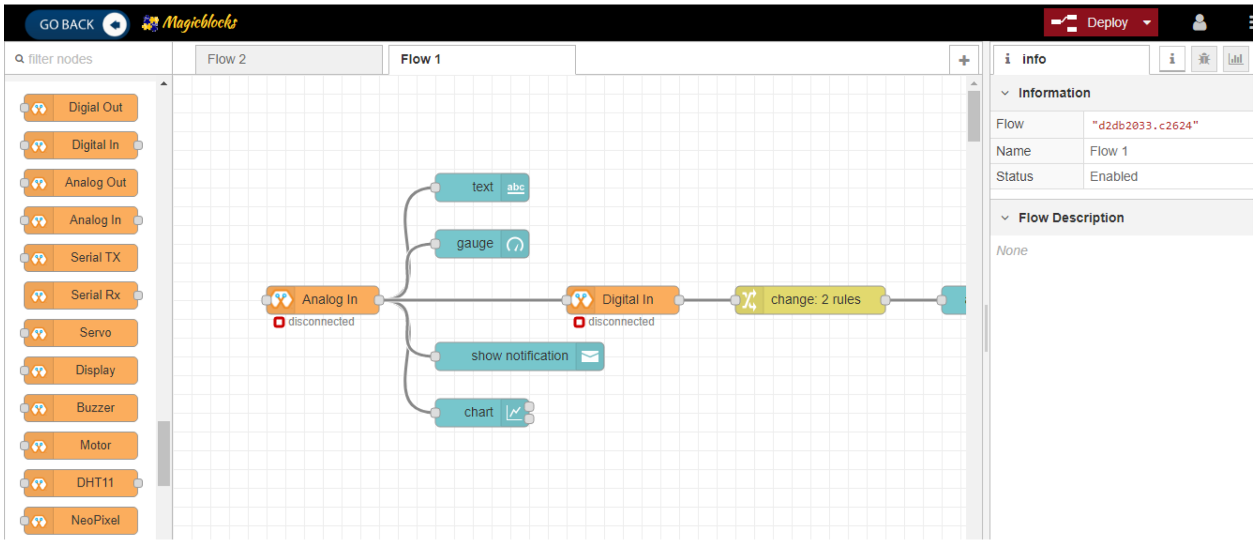
- Once saved, the Magicbit will restart and try to connect to the WiFi and then to magicblocks. You can connect to the Magicbit's configuration WiFi AP and navigate to http://192.168.4.1 to check the status of the connection.
- Login to magicblocks, navigate to **Device Manager** and check if the **Connection** column comes up as a blinking green indicator. If yes, you can proceed to the next step

4.6 Go to the playground!

- Login to magicblocks and click on **Edit** to open the playground.



- The playground is a hosted instance of a customized version of the open source Node-Red application. On the left you will find the palette where all the blocks will reside. You can drag and drop the blocks from the palette to the canvas and start rolling very quickly. Once done click the **Deploy** button on the top right corner and all your changes will be saved. In the next section we will go through some examples covering all the relevant blocks



CHAPTER 5

Hello Magicbit

Let's start magic with displaying a text on Magicbit display.

Open the Device Manager in your magicblocks account

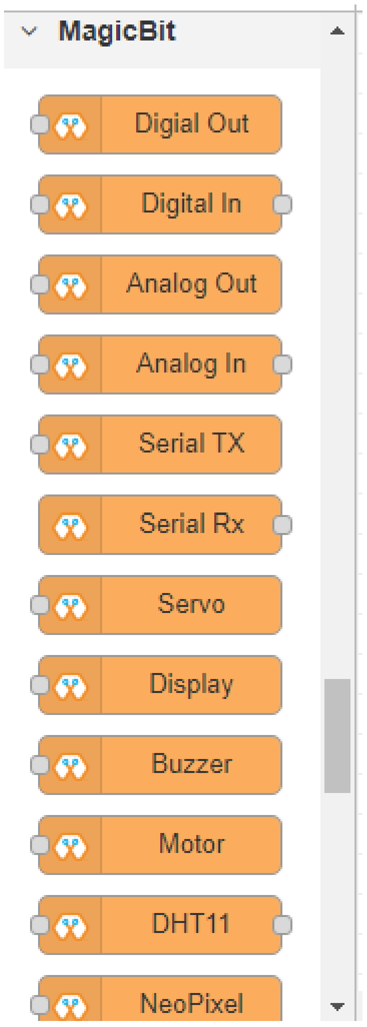
- Turn on your Magicbit board that was setup in the previous section and wait for it to connect
- Copy the device Id of this Magicbit board
- Open the playground
- Drag and drop the **Display** block under the category **Magicbit**
- Double click on the block and paste the device Id
- Drag and drop an **inject** blocks under the **input** category and connect them to the **Display** block
- In one **inject** node set payload type as String and type Hello Magicbit in the text box. You can set a topic too.
- Click deploy
- Click button in Inject node to see the magic. You can set any text from any where in the world!!!

Watch on Youtube https://www.youtube.com/watch?v=6y6Ieq8vZbo

Warning: If you are not familiar with Magicblocks or Nodered you can quickly learn essential features from here. https://magicbit-magicblocksio.readthedocs.io/en/latest/#playground

CHAPTER 6

Magicbit Blocks



Following Blocks are available.

- Digital Out
- Digital In
- Analog Out/PWM
- Analog In
- Serial Tx
- Serial Rx
- Servo
- Display
- Buzzer
- Motor
- DHT11
- NeoPixel
- Ultrasonic

This block set enables you to control individual pins of the device from the playground. The functionality of each block is described below:

6.1 How Configure Blocks

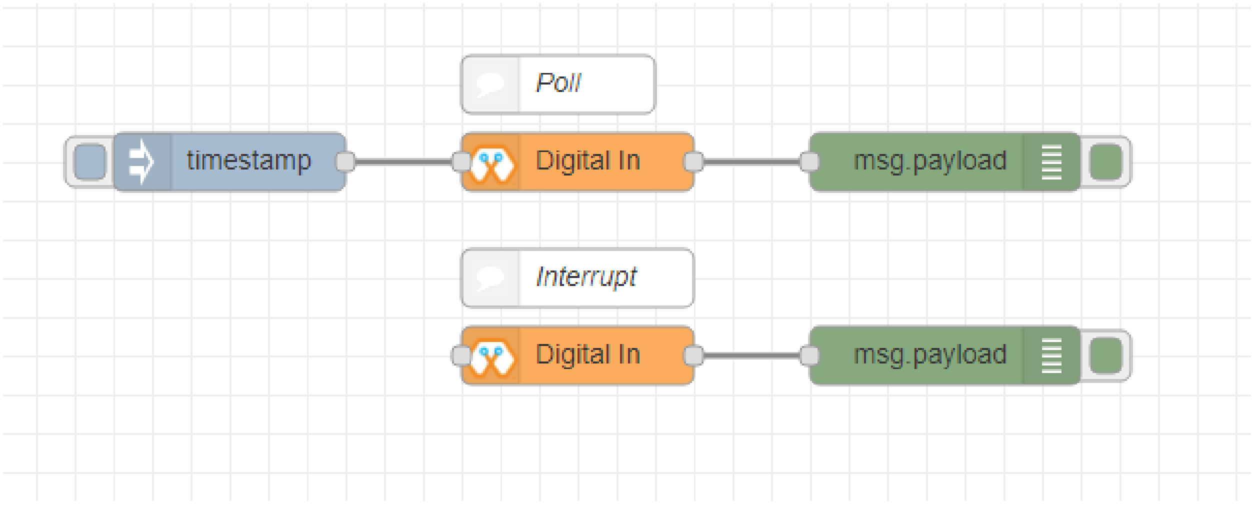
Every block has a property called Device ID where you need to specify to which device this block belongs to. This is important because you will be working with multiple devices in a typical IoT project. You can find device ID from magicblocks device manager.

6.1.1 Digital Out

This block is used to set a digital output pin to 1 or 0 based on the input. An input of 1 or true will make the configured pin go HIGH and vice versa

- **Configuration**
  - Pin: pin number of the Digital pin to write to. Available pins can be selected from dropdown list.
  - Name: Any name desired
- **Input**
  - value to be written to the pin. Accepts 1 (true) or 0 (false) eg: {“payload”: 1}

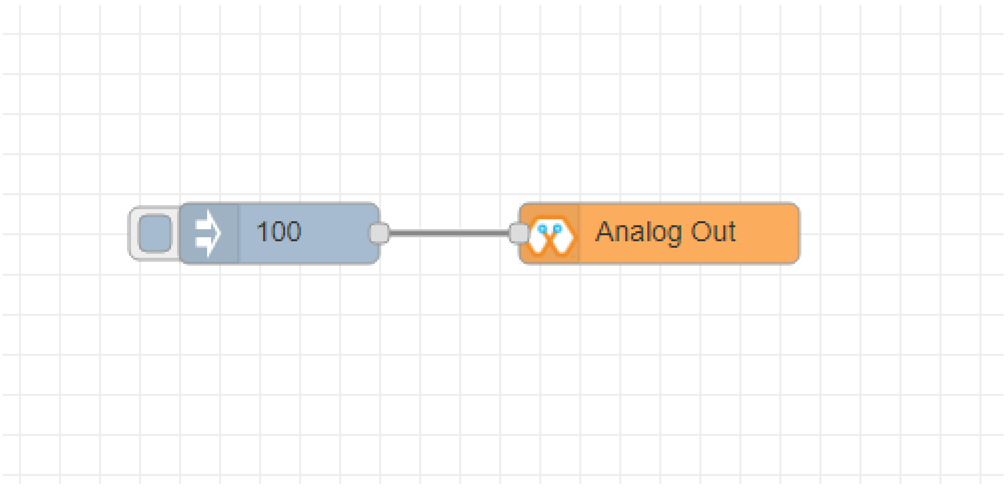
6.1.2 Digital In



This block will read Input status of a pin. There are two methods to get input 1. Poll - Block need to triggered to get input status. Any input will serve as a trigger. (eg:inject block) 2. Interrupt - If there is any change of pin state of Magicbit block will output the current state Input status can be passed to a another block or viewed on the debug window.

- **Configuration:**
  - Pin: pin number of the digital pin to read .Select from drop down list
  - Name: Any name desired
  - Method: Poll/Interrupt
- **Input**
  - Any input. Used as a trigger
- **Output**
  - Value of the pin as 1 or 0 in the following format and the pin number as the topic

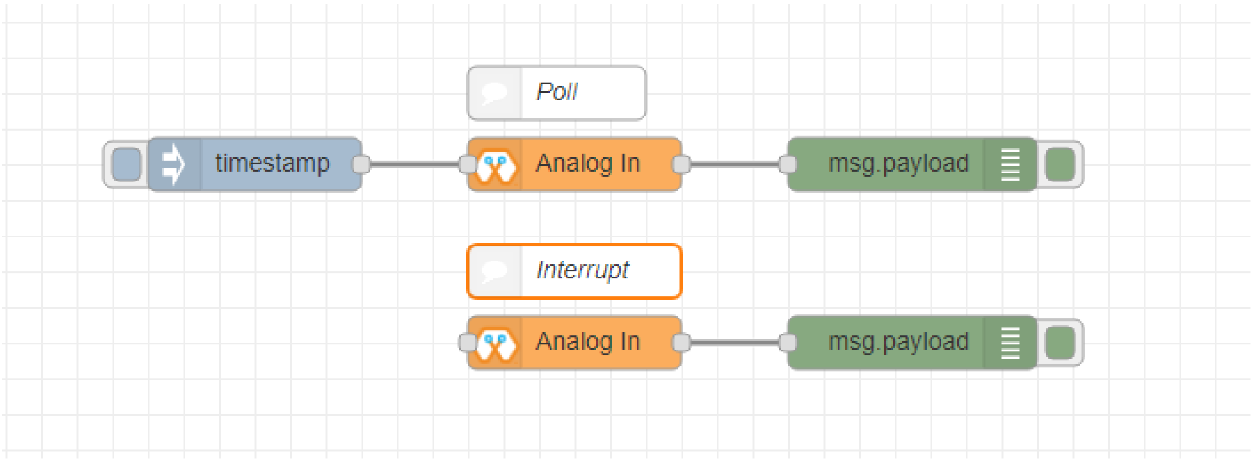
6.1.3 Analog Out



This block used to set PWM to pins of Magicbit. Value should be in range of 0-255. Inject block or output of a another block can be used to set the value.

- **Configuration:**
  - Pin: pin number of the to set PWM. Select from drop down list
  - Name: Any name desired
- **Input**
  - Inject block or any block. Input value should be in range of 0-255

6.1.4 Analog In

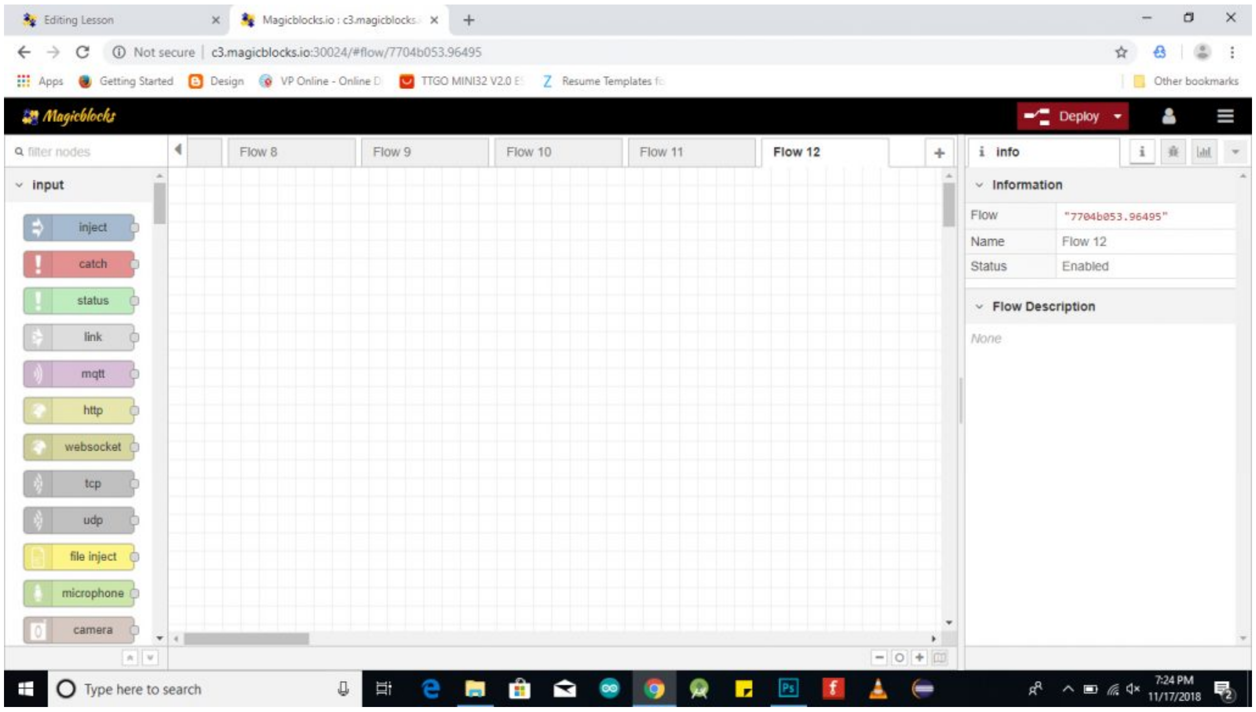


This block will read analog value of the ADC pin of the module. Similar to the digital in block you need to set method to read the value. Any input sent to the block will serve as the trigger. 1. Poll - Block need to triggered to get input status. Any input will serve as a trigger. (eg:inject block) 2. Interrupt - If there is any change greater or less than threshold value of Magicbit block will output the analog value

- **Configuration:**
  - Pin: pin number of the analog pin to read (Required)
  - Name: Any name desired
  - Method: Poll/Interrupt
  - Threshold: If interrupt method selected value return from output if there is any change greater or less than this value
- **Input**
  - Any input. Used as a trigger
- **Output**
  - Value of the pin from 0 to 4096 (12bit ADC) {“payload”: 965}

CHAPTER 7

Playground



The editor window consists of four components:

- The header at the top, containing the deploy button, main menu, and, if user authentication is enabled, the user menu.
- The palette on the left, containing the nodes available to use.
- The main workspace in the middle, where flows are created.
- The sidebar on the right.

The main workspace is where flows are developed by dragging nodes from the palette and wiring them together. The workspace has a row of tabs along the top; one for each flow and any subflows that have been opened.

7.1 Flow

7.1.1 Adding a flow

To add a new flow, click the .. image:: https://github.com/magicbitlk/Magicbit-Magicblocks.io/blob/master/Images/plus.png?raw=true

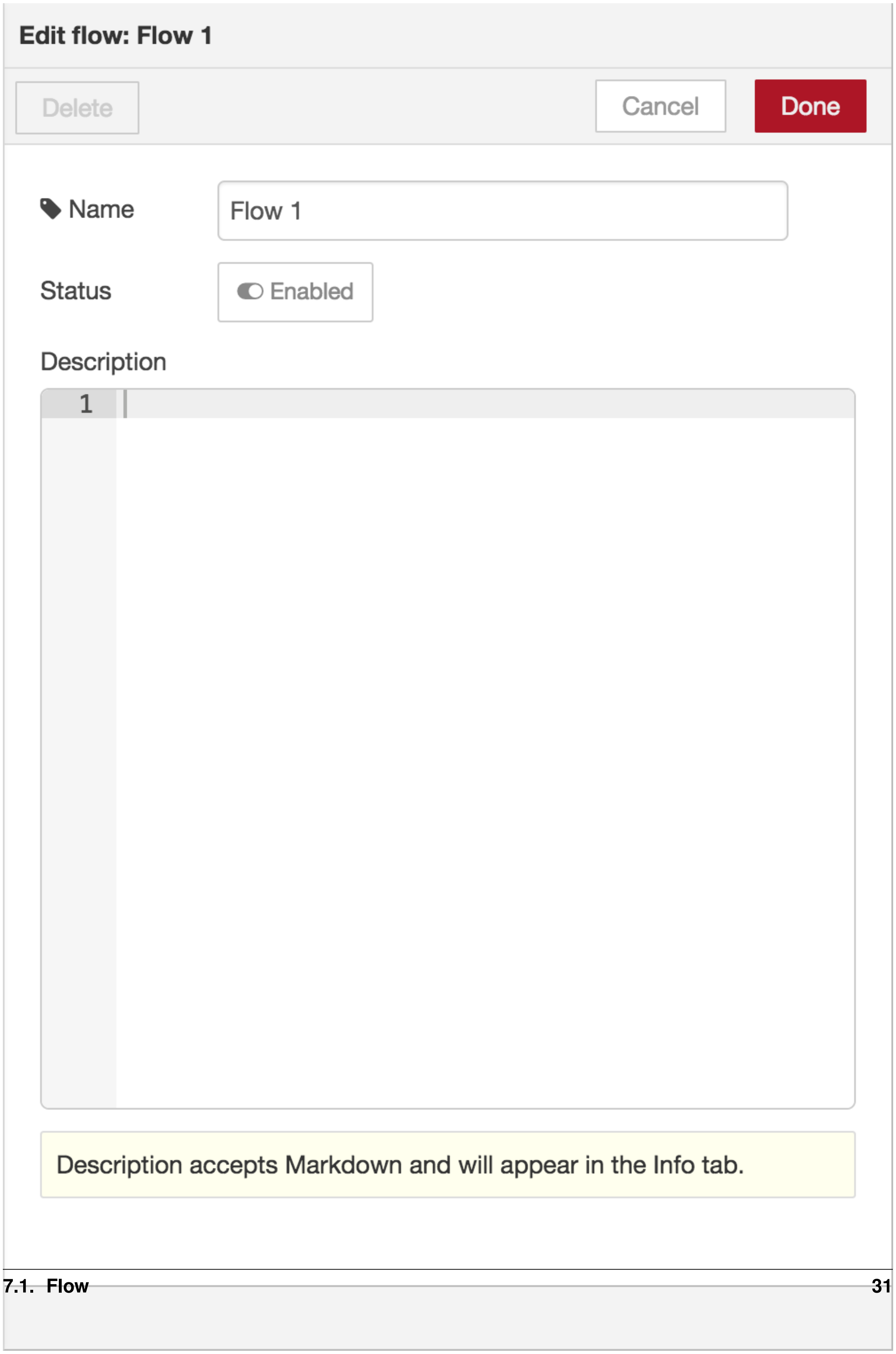
button in the top bar.

7.1.2 Editing flow properties

To edit a flow's properties, double-click on its tab in the top bar. This will open the Flow Properties dialog.

Within the dialog, the flow's name and description can be set. The description can use Markdown syntax for formatting and will appear in the Information sidebar.

The Status property can be used to disable or enable the flow.



Name: Flow 1

Status: Enabled

Description

Description accepts Markdown and will appear in the Info tab.

7.1.3 Deleting a flow

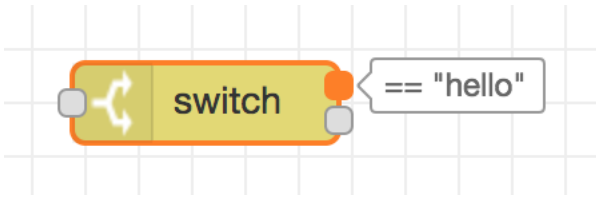
To delete a flow, click the 'Delete' button in the Flow Properties dialog.

7.2 Nodes

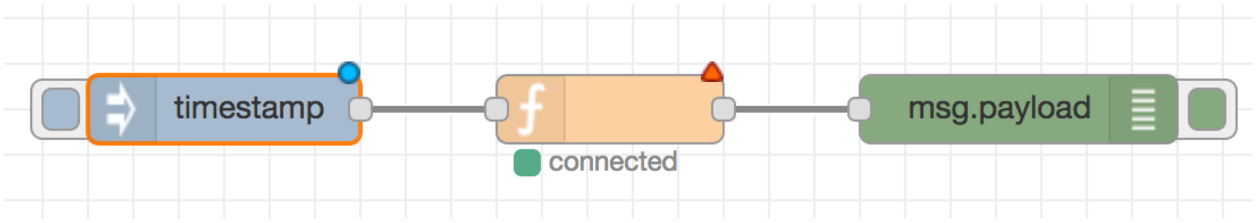
Nodes can be added to the workspace by either:

- Dragging them from the palette
- Using the quick-add dialog
- Importing from the library or clipboard

Nodes are joined together by wires via their ports. A node can have at most one input port and many output ports. A port may have a label that is displayed when the mouse hovers over it. A node may specify labels, for example, the Switch node shows the rule that matches the port. The labels can also be customised in the node edit dialog.



Some nodes display a status message and icon below the node. This is used to indicate the runtime state of the node - for example, the MQTT nodes indicate if they are currently connected or not.

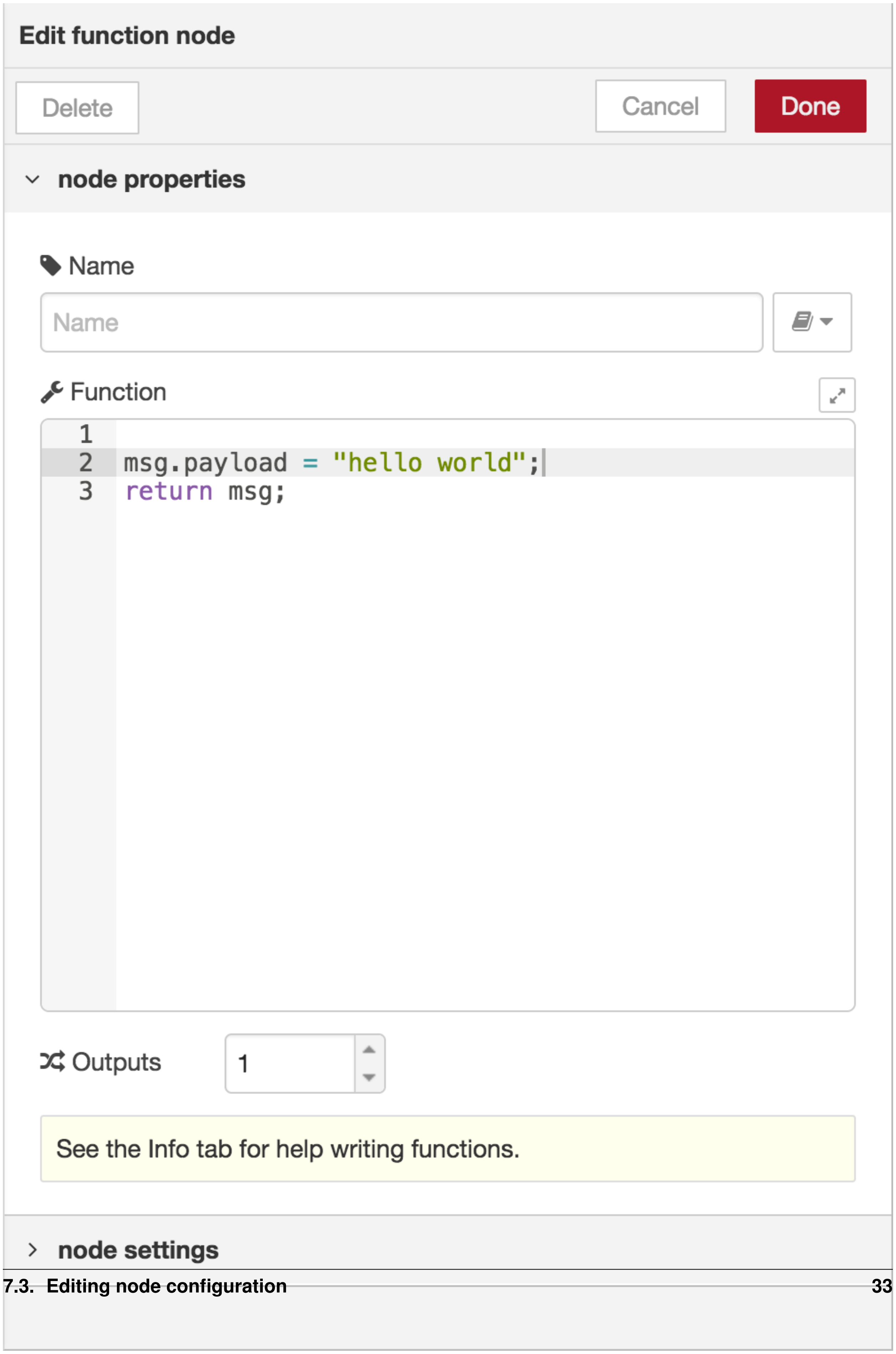


If a node has any undeployed changes, it displays a blue circle above it. If there are errors with its configuration, it displays a red triangle.

Some nodes include a button on either its left or right edge. These allow some interaction with the node from within the editor. The Inject and Debug nodes are the only core nodes that have buttons.

7.3 Editing node configuration

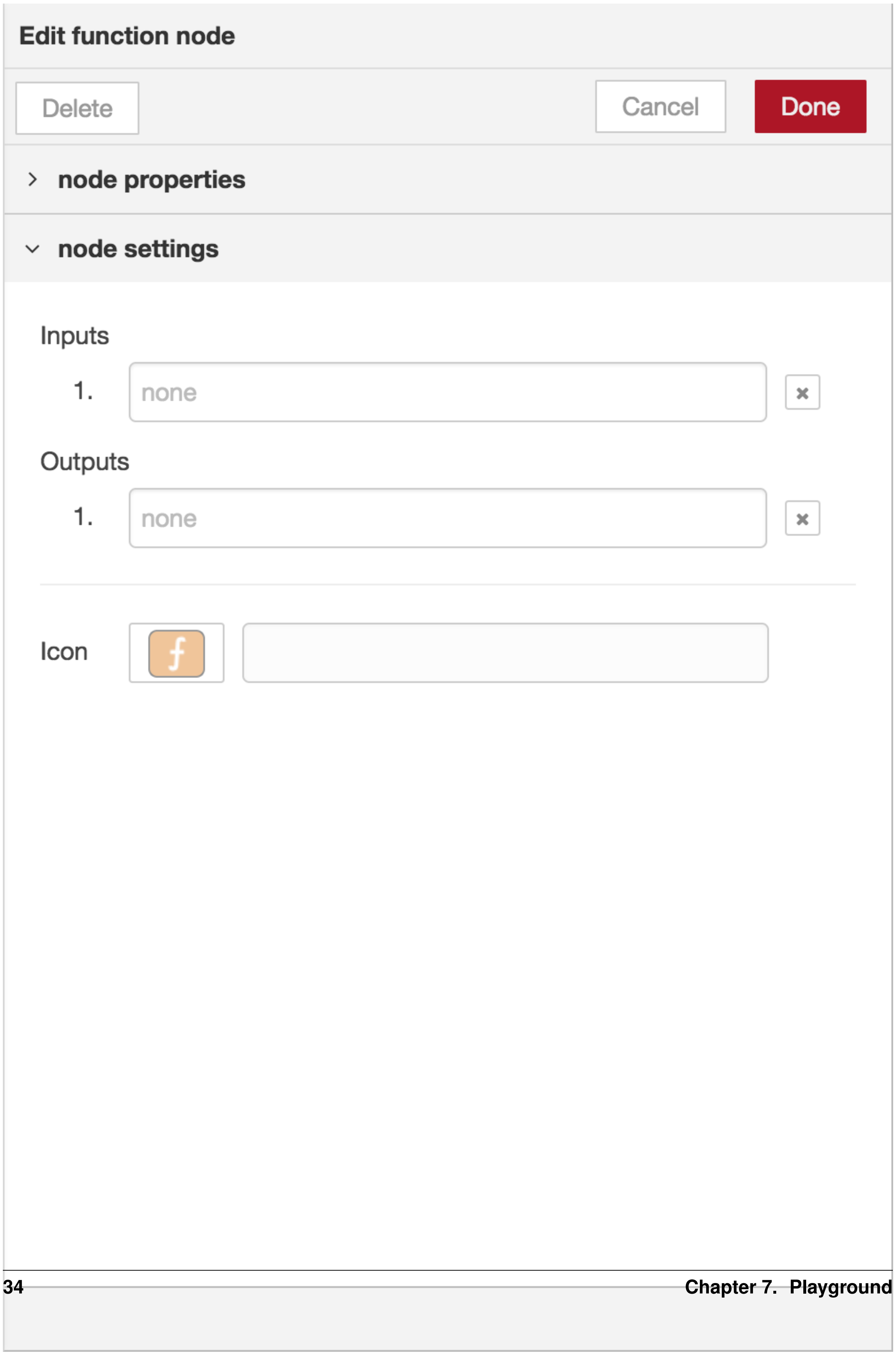
A node's configuration can be edited by double clicking on the node, or pressing **Enter** when the workspace has focus. If multiple nodes are selected, the _first_ node in the selection will be edited.



```
msg.payload = "hello world";
return msg;
```

Outputs: 1

See the Info tab for help writing functions.


Edit function node
Delete
Cancel
Done
node properties
node settings
Inputs
1. none
Outputs
1. none
Icon

The node edit dialog has two separate sections; properties and settings. The properties section shows the edit form specific to the node type being edited. The settings section shows the common settings that can be set on all nodes. This includes the custom port labels as well as the icon for the node.

Clicking on the icon shows the Node icon picker that can be used to select the icon for the node from the list of all available icons.

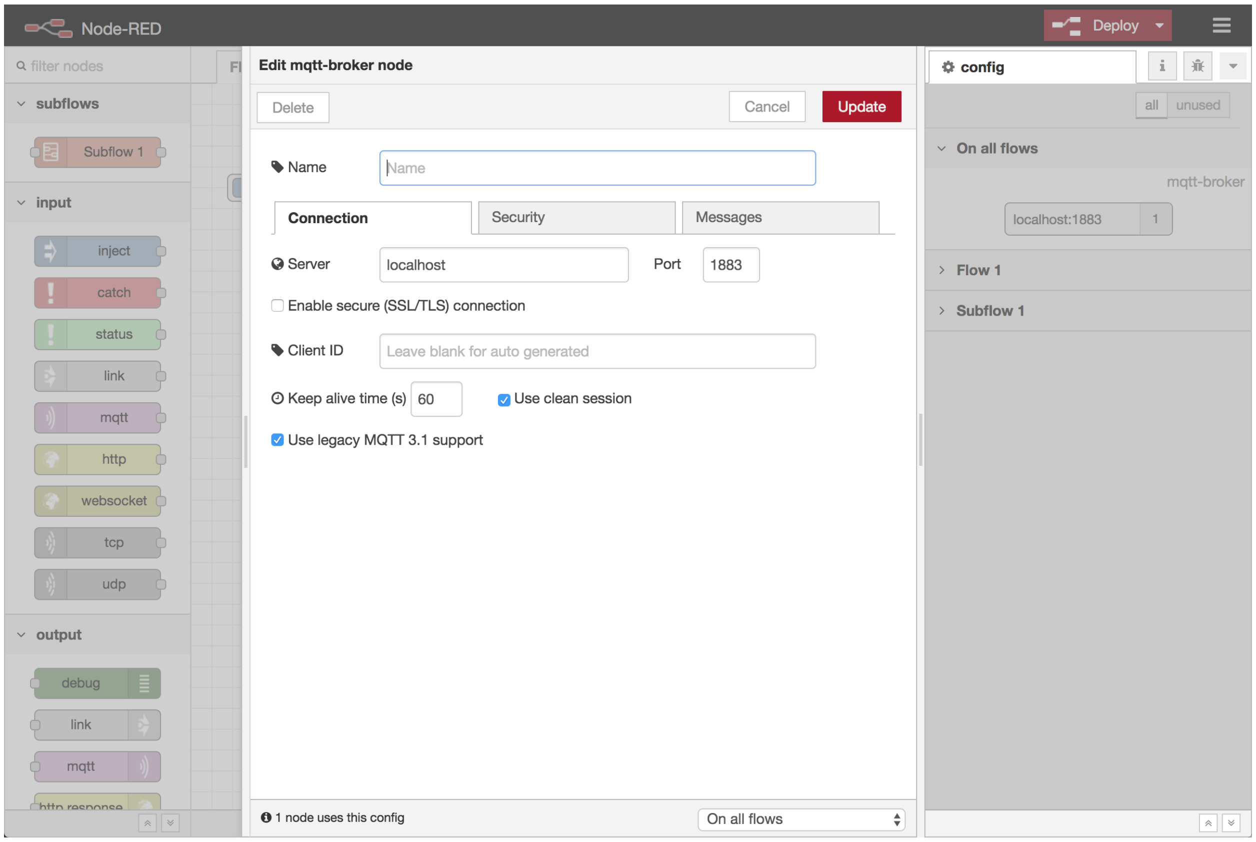
7.3.1 Configuration nodes

A Configuration (config) Node is a special type of node that holds reusable configuration that can be shared by regular nodes in a flow.

For example, the MQTT In and Out nodes use an MQTT Broker config node to represent a shared connection to an MQTT broker.

Configuration nodes are added through the edit dialog of a node that requires the config node. It will have a field to select from the available config nodes of the required type or to add a new instance. .. image:: https://github.com/magicbitlk/Magicbit-Magicblocks.io/blob/master/Images/editor-edit-node-config-node.png?raw=true

Clicking the button next to the select box will open the edit dialog for the selected node, or add a new instance.


Edit mqtt-broker node

Name: Name

Connection | Security | Messages

Server: localhost Port: 1883

☐ Enable secure (SSL/TLS) connection

Client ID: Leave blank for auto generated

Keep alive time (s): 60 ☑ Use clean session

☑ Use legacy MQTT 3.1 support

1 node uses this config On all flows

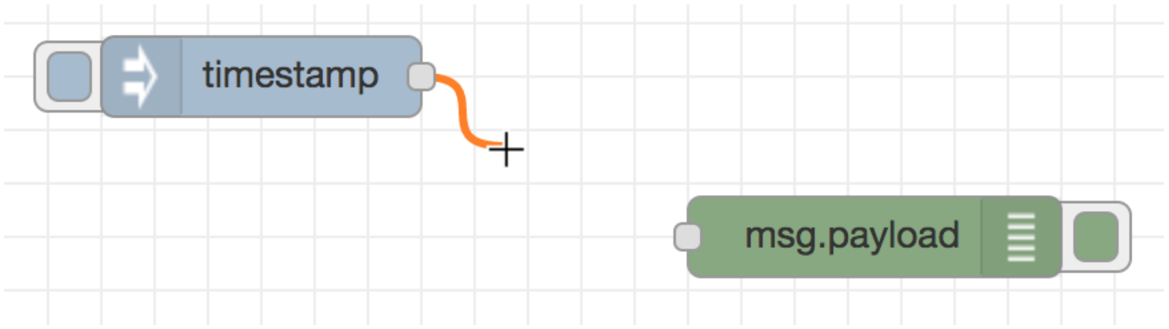
The config node edit dialog only has the node properties section - as a config node has no icon or ports to set labels on.

In the footer of the dialog is an indication of how many nodes use this config node. It also provides a select box to set the scope of the config node. The scope determines which flows the config node is available on. By default it is available on all flows, but the select box can be used to make it local to just one flow.

The Configuration Nodes Sidebar can be used to manage all config nodes.

7.4 Wires

Nodes are wired together by pressing the left-mouse button on a node's port, dragging to the destination node and releasing the mouse button.

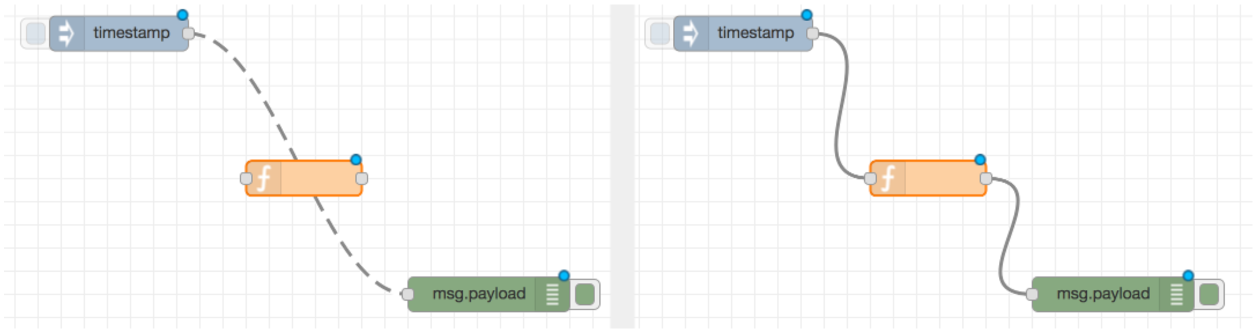


Alternatively, if the **Ctrl/Command** key is held down, the left-mouse button can be clicked (and released) on a node's port and then clicked on the destination. If the **Ctrl/Command** key remains held and the just-wired destination node has an output port, a new wire is started from that port. This allows a set of nodes to be quickly wired together.

This can also be combined with the Quick-Add dialog that is triggered by a **Ctrl/Command-Click** on the workspace to quickly insert new nodes and have them already wired to previous nodes in the flow.

7.4.1 Splitting wires

If a node with both an input and output port is dragged over the mid-point of a wire, the wire is draw with a dash. If the node is then dropped, it is automatically inserted into the flow at that point.



7.4.2 Moving wires

To disconnect a wire from a port, select the wire by clicking on it, then press and hold the **Shift** key when the left-mouse button is pressed on the port. When the mouse is then dragged, the wire disconnects from the port and can be dropped on another port. If the mouse button is released over the workspace, the wire is deleted.

If a port has multiple wires connected to it, if none of them is selected when button is pressed with the **Shift** key held, all of the wires will move.

7.4.3 Deleting wires

To delete a wire, first select it by clicking on it and then press the **delete** key.

7.5 Selection

A node is selected when it is clicked on. This will deselect anything currently selected. The Information Sidebar will update to show the node's properties and help text for its type.

If the **Ctrl** or **Command** key is held when clicking on the node, the node will be added to the current selection (or removed if it was already selected).

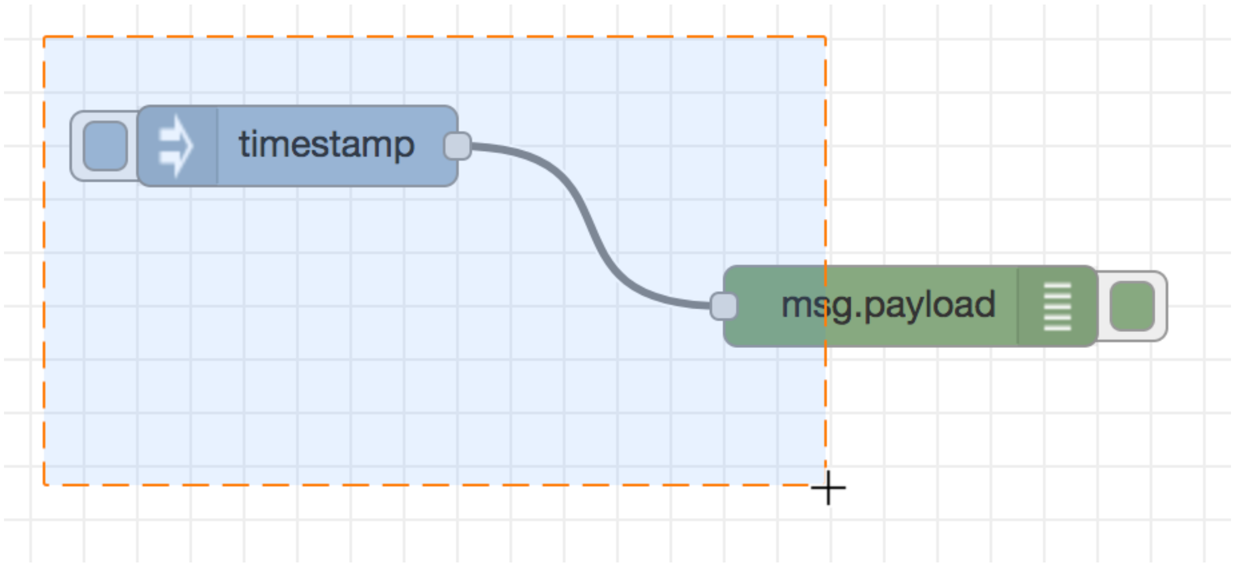
If the **Shift** key is held when clicking on the node, it will select that node and all other nodes it is connected to.

A wire is selected when it is clicked on. Unlike nodes, it is only possible to select one wire at a time.

7.5.1 Lasso Tool

The lasso tool can be used to select multiple nodes. It is enabled by click-dragging on the workspace.

It cannot be used to select a wire.



7.5.2 Selecting all nodes

To select all nodes on the current flow, ensure the workspace has focus and then press **Ctrl/Command-a**.

7.5.3 Editor clipboard

The editor supports the standard copy/cut/paste actions. Note they use an internal clipboard rather than the system clipboard.

7.6 Import & Export

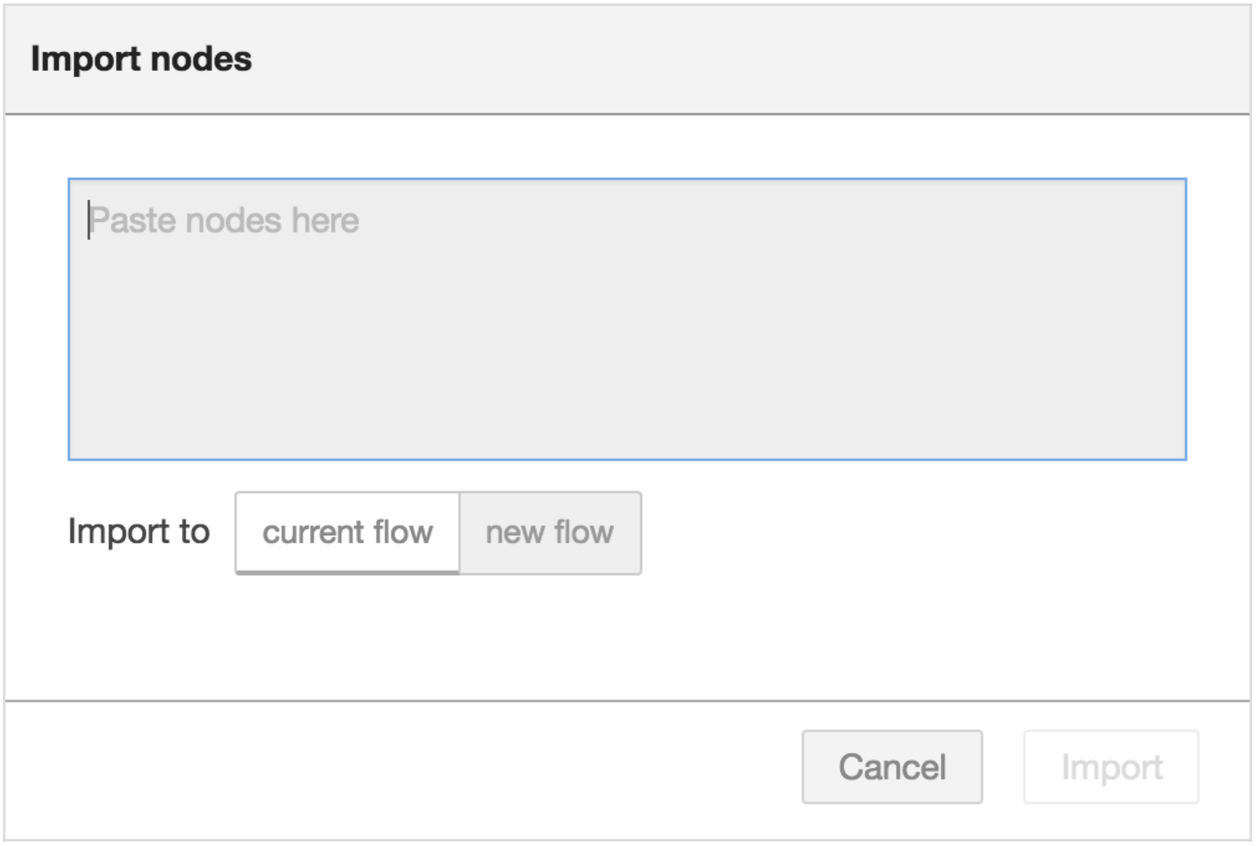
Flows can be imported and exported from the editor using their JSON format, making it very easy to share flows with others.

7.6.1 Importing flows

To import a flow, open the Import dialog, paste in the flow json and click ‘Import’.

The ‘Import’ button will only be active if valid JSON is pasted into the dialog.

The dialog also offers the option to import the nodes into the current flow, or to create a new flow for them.

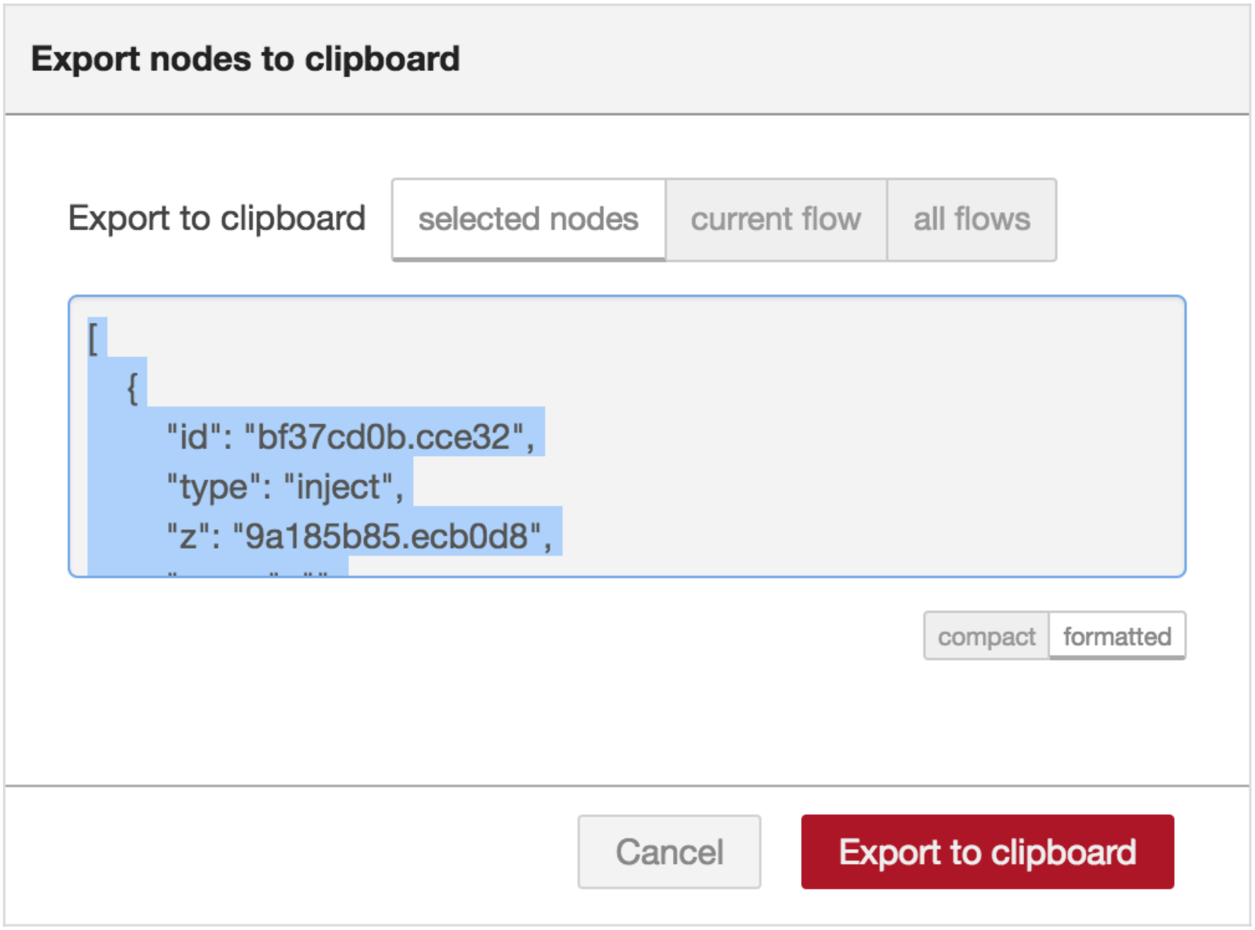


7.6.2 Exporting flows

The Export dialog can be used to copy flow json out of the editor.

It can export either the selected nodes, the current flow (including its tab node) or the complete flow configuration.

It offers the option to export compact or formatted JSON. The compact option generates a single line of JSON with no whitespace. The formatted JSON option is formatted over multiple lines with full indentation - which can be easier to read.



Export nodes to clipboard

Export to clipboard

```
[
    {
        "id": "bf37cd0b.cce32",
        "type": "inject",
        "z": "9a185b85.ecb0d8",
```

CHAPTER 8

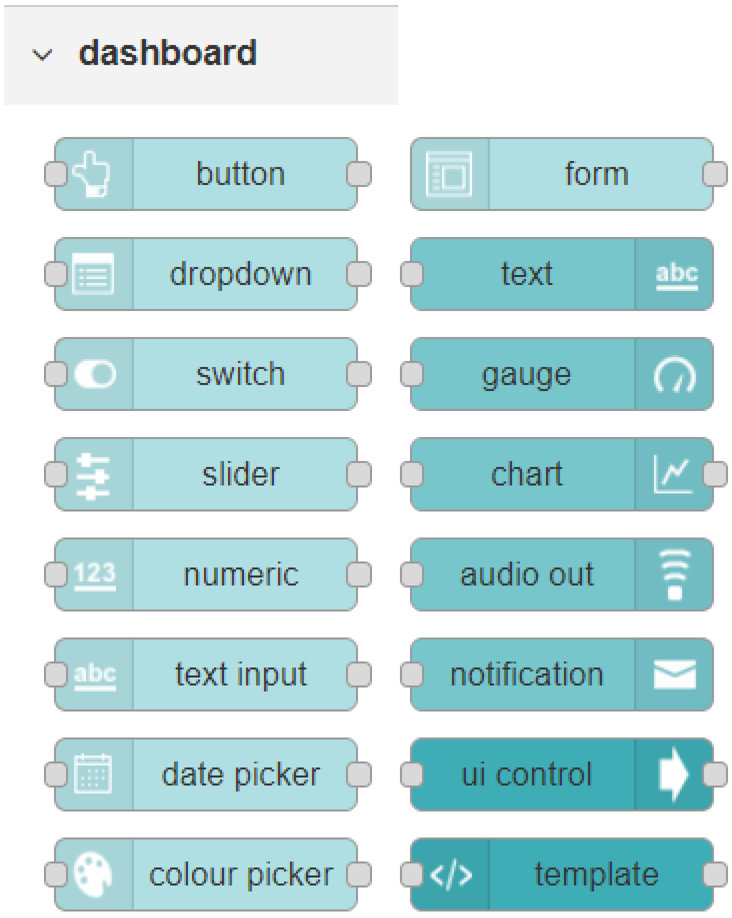
Dashboards

8.1 What is Magicblocks Dashboard?

Magicblocks Dashboard is a module that provides a set of nodes in **Magicblocks** to quickly create a live data dashboard.

Since **Magicblocks** is based on NODE-RED opensource platform you can learn more about dashboard using following links

- http://flows.nodered.org/node/node-red-dashboard
- https://github.com/node-red/node-red-dashboard



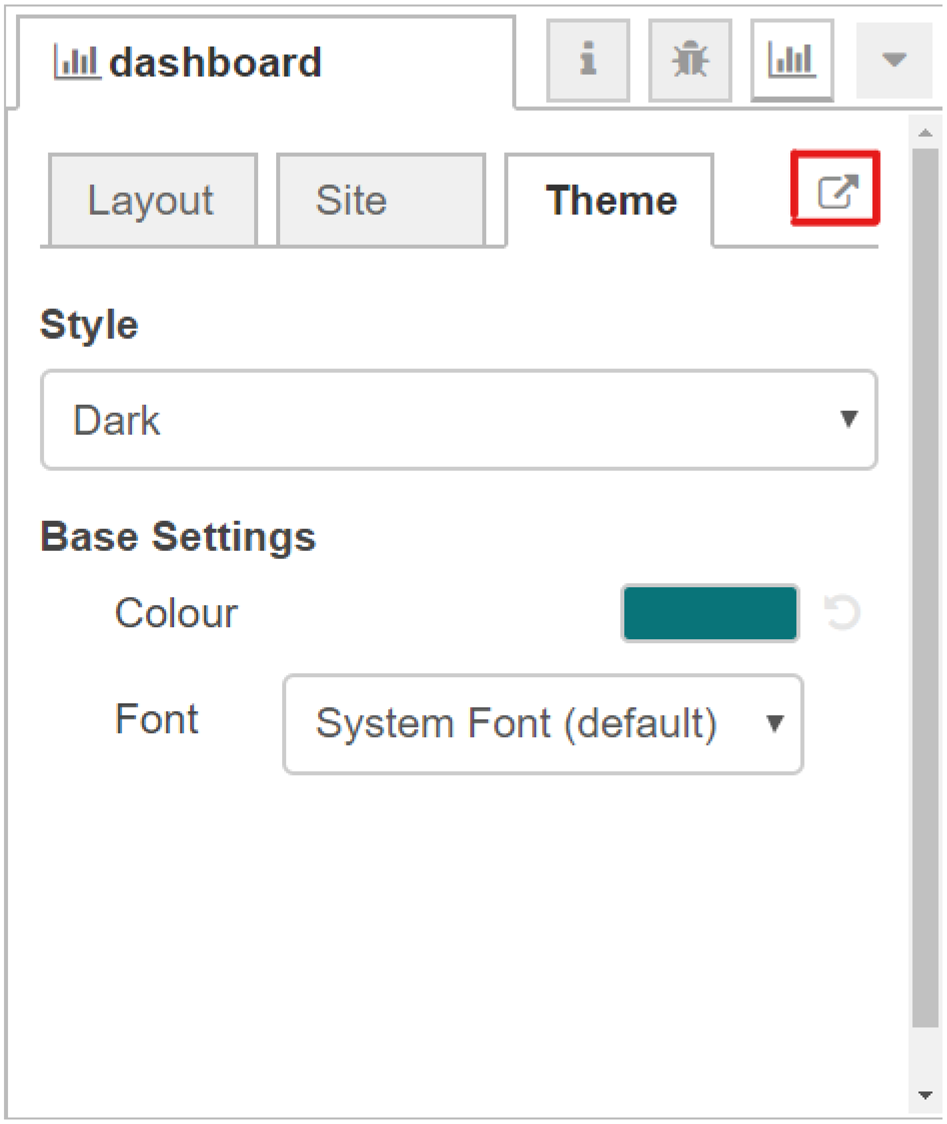
Nodes from the dashboard section provide widgets that show up in your application user interface (UI).

The user interface is organized in tabs and groups. Tabs are different pages on your user interface, like several tabs in a browser. Inside each tab you have groups that divide the tabs in different sections so that you can organize your widgets.

Every widget should have an associated group that determines where the widget should appear on the user interface.

To create a tab and a group follow the following instructions (see figure below):

- On top right corner of the **Magicblocks** window you have a tab called dashboard.
- Select that tab **(1)**. To add a tab to the user interface click on the +tab button **(2)**.
- Once created, you can edit the tab by clicking on the edit button **(3)**



Style: Dark

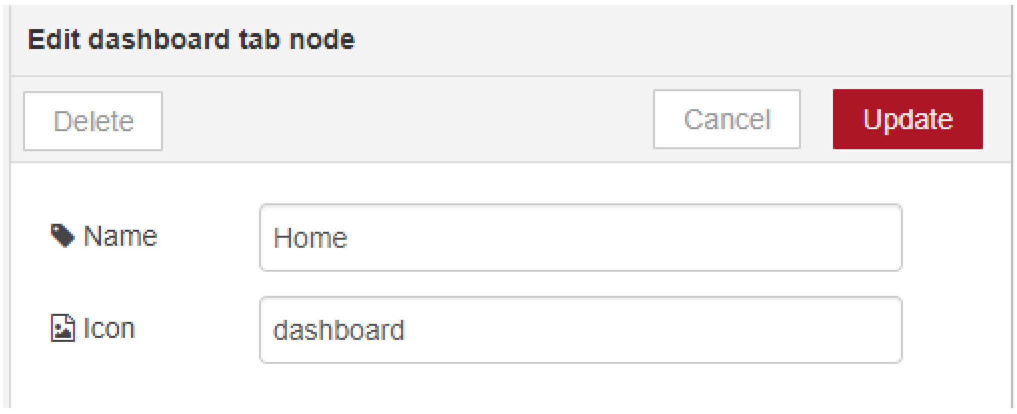
Base Settings

Colour

Font: System Font (default)

You can edit the tab's name and change its icon

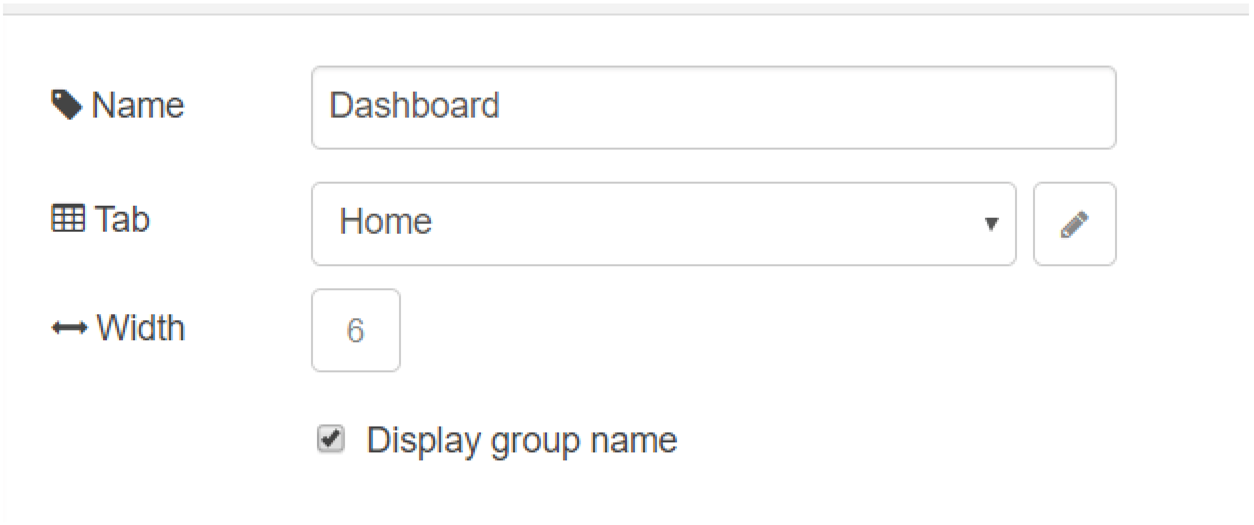
- Name: you can call it whatever you want
- Icon: you should use a name accordingly to the icon's names in this link: https://klarsys.github.io/angular-material-icons



Name: Home

Icon: dashboard

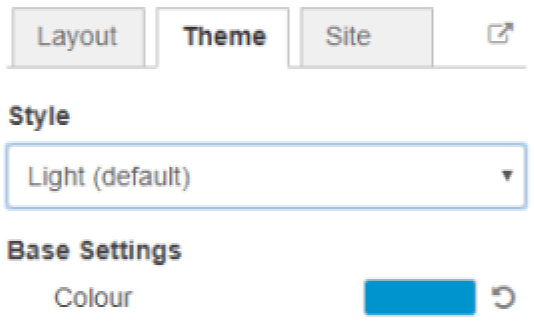
- After creating a tab, you can create several groups under that tab. You need to create at least one group to add your widgets. To add a group to the created tab, you need to click on the +group button **(4)**.
- Then, you can edit the created group by clicking on the edit button **(5)**.


Name: Dashboard
Tab: Home
Width: 6
Display group name: ✓

- You can edit its name, select its corresponding tab and change its width.

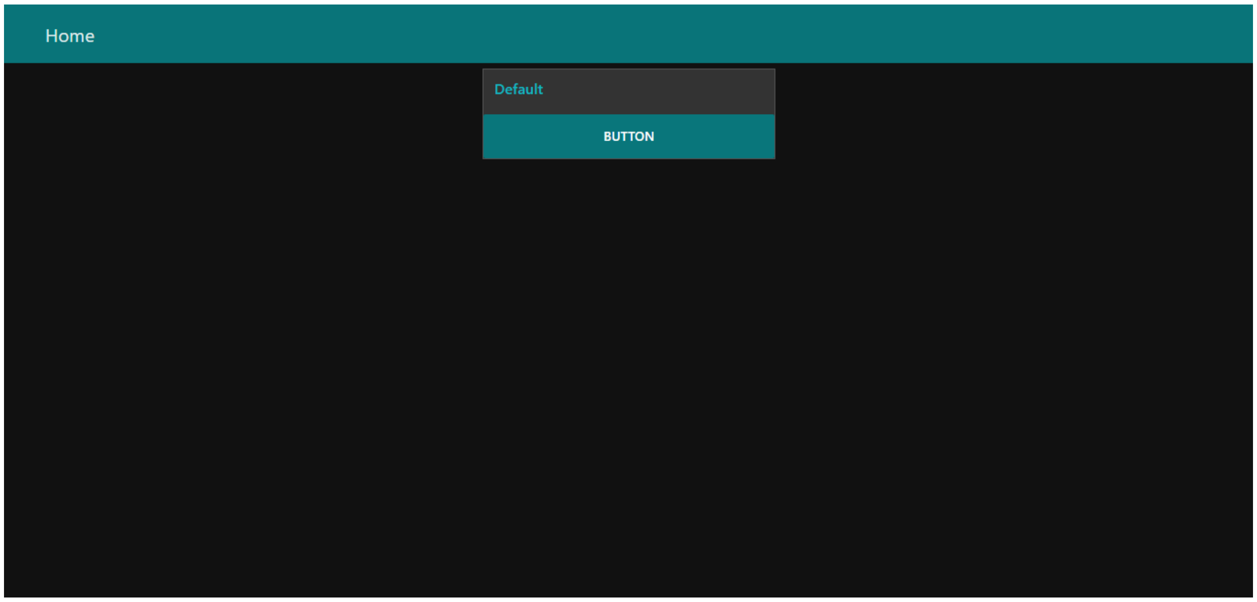
8.2 Dashboard Theme

The **Magicblocks** Dashboard has a white background and a light blue bar by default. You can edit its colors in the Theme tab on the up right corner as show in the following figure.



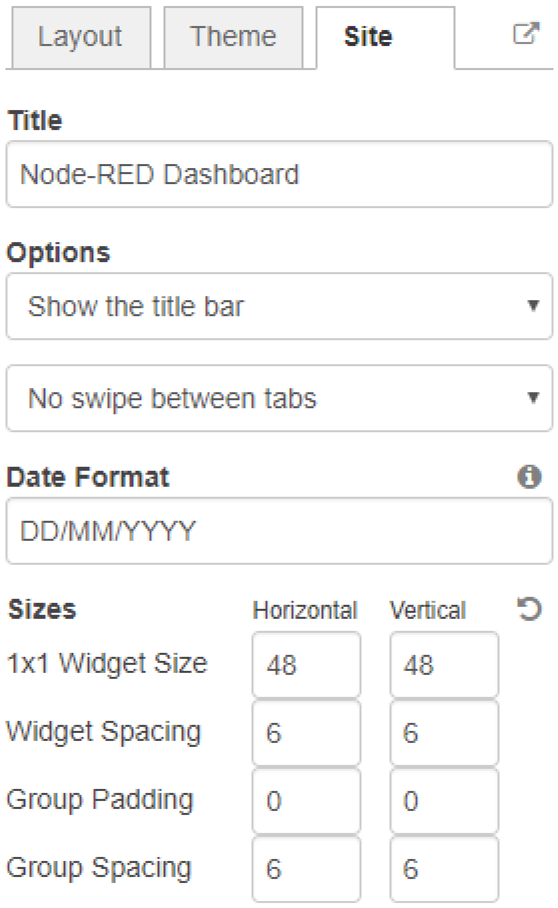
Style: Light (default)
Base Settings
Colour

- Change the style, deploy the changes and see the Dashboard UI changing its colors. For example, like in the following figure



8.3 Dashboard Site

At the right upper corner of the **Magicblocks** window, you have another tab called Site that allows you to do further customization as show in the figure below.



Title
Node-RED Dashboard

Options
Show the title bar
No swipe between tabs

Date Format
DD/MM/YYYY

Sizes	Horizontal	Vertical
1x1 Widget Size	48	48
Widget Spacing	6	6
Group Padding	0	0
Group Spacing	6	6

Feel free to change the settings, then deploy the changes and see how the UI looks. At the moment you won't see much difference because you haven't added anything to the dashboard yet. Those changes will be noticeable when you start adding widgets to the UI.

8.4 Creating a User Interface – Example

In this section we're going to make a dashboard example to show you how you can build and edit your own dashboard – we won't actually add functionalities to the widgets – we'll do that in future projects. This dashboard will have the following features

- Two different tabs: one called Room and another called Garden
- The Room tab will have two groups and the Garden tab will have one group
- We'll add a color picker and a switch to the room groups
- We'll add a chart to the Garden group

8.5 Creating the Tabs

On the top right corner on the Magicblocks window, select the **dashboard** tab and create two new tabs by clicking on the **+tab** button.

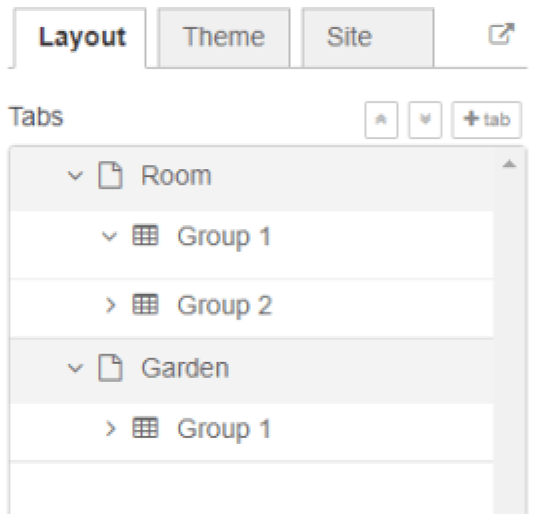
Edit one tab with the following properties

- **Name:** Room
- **Icon:** tv

And the other one with the following

- Name: Garden
- Icon: local_florist

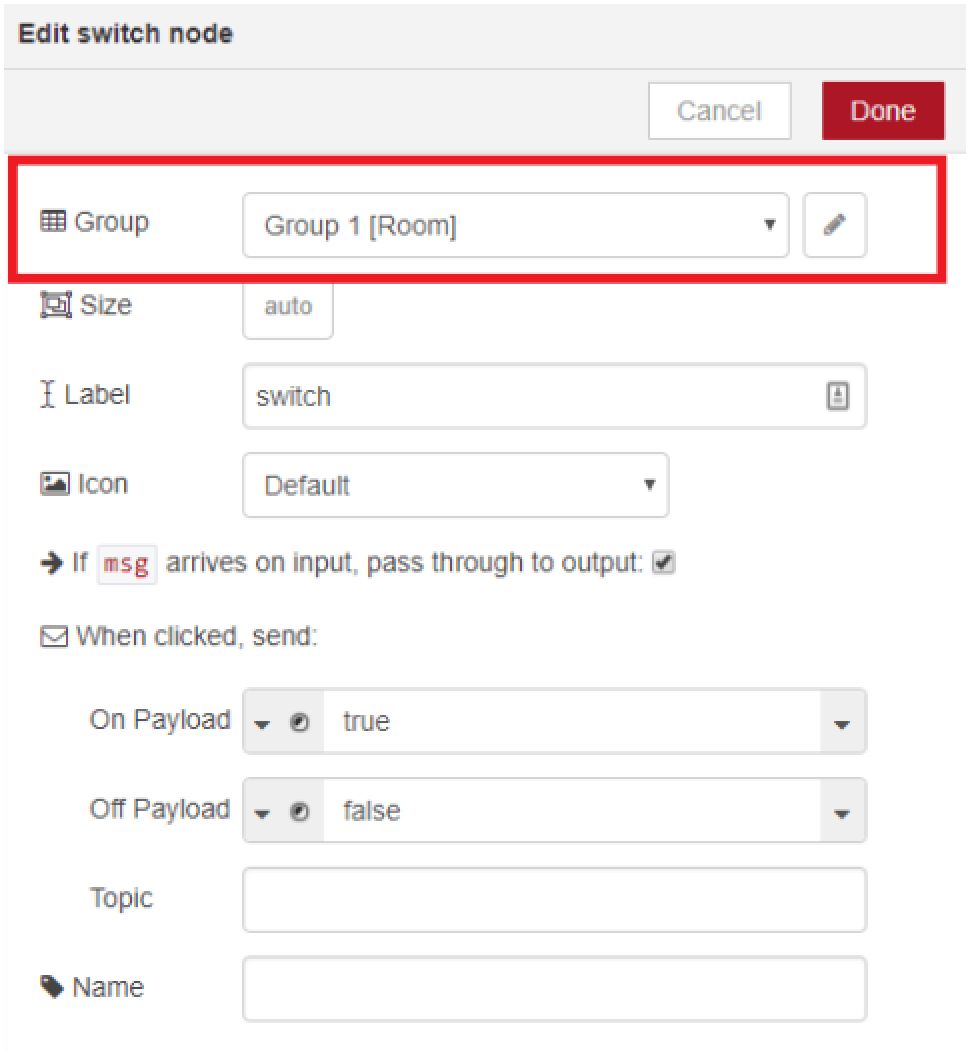
Then, add two groups to the Room tab and one group to the Garden tab. The following figure shows how your dashboard layout looks.



8.6 Adding the Widgets

Add a switch, a slider, a colour picker and a gauge to the flow as show in the following figure .. image:: https://github.com/magicbitlk/Magicbit-Magicblocks.io/blob/master/Images/flow.png?raw=true

Double click on the switch. A new window pops up.


Edit switch node

Group: Group 1 [Room]
Size: auto
Label: switch
Icon: Default
If msg arrives on input, pass through to output: ✓
When clicked, send:
On Payload: true
Off Payload: false
Topic:
Name:

In this new window you can choose where you want your button widget to appear. In this case we want it to appear in the Room tab, Group 1 as highlighted in red in the previous figure.

Then, do the same for the other widgets but add them to the following groups:

- **slider:** Group 1 [Room]
- **color picker:** Group 2 [Room]
- **gauge:** Group 1 [Garden]

Source: (https://randomnerdtutorials.com/getting-started-with-node-red-dashboard/)

CHAPTER 9

Core Blocks

The Magicblocks palette includes a default set of nodes that are the basic building blocks for creating flows. This page highlights the core set you should know about.

All nodes include documentation you can see in the Info sidebar tab when you select a node.

- Inject
- Debug
- Function
- Change
- Switch
- Template

9.1 Inject node

The Inject node can be used to manual trigger a flow by clicking the node's button within the editor. It can also be used to automatically trigger flows at regular intervals.

The message sent by the Inject node can have its **payload** and **topic** properties set.

The **payload** can be set to a variety of different types:

- a flow or global context property value
- a String, number, boolean, Buffer or Object
- a Timestamp in milliseconds since January 1st, 1970

9.2 Debug node

The Debug node can be used to display messages in the Debug sidebar within the editor.

The sidebar provides a structured view of the messages it is sent, making it easier to explore the message.

Alongside each message, the debug sidebar includes information about the time the message was received and which Debug node sent it. Clicking on the source node id will reveal that node within the workspace.

The button on the node can be used to enable or disable its output. It is recommended to disable or remove any Debug nodes that are not being used.

The node can also be configured to send all messages to the runtime log, or to send short (32 characters) to the status text under the debug node.

The page on Working with messages gives more information about using the Debug sidebar.

9.3 Function node

The Function node allows JavaScript code to be run against the messages that are passed through it.

A complete guide for using the Function node is available here .

9.4 Change node

The Change node can be used to modify a message's properties and set context properties without having to resort to a Function node.

Each node can be configured with multiple operations that are applied in order. The available operations are:

- **Set** - set a property. The value can be a variety of different types, or can be taken from an existing message or context property.
- **Change** - search and replace parts of a message property.
- **Move** - move or rename a property.
- **Delete** - delete a property.
- **When** setting a property, the value can also be the result of a JSONata expression. JSONata is a declarative query and transformation language for JSON data.

9.5 Switch node

The Switch node allows messages to be routed to different branches of a flow by evaluating a set of rules against each message.

The node is configured with the property to test - which can be either a message property or a context property.

There are four types of rule:

- **Value** rules are evaluated against the configured property
- **Sequence** rules can be used on message sequences, such as those generated by the Split node
- A JSONata **Expression** can be provided that will be evaluated against the whole message and will match if the expression returns a _true_ value.
- An **Otherwise** rule can be used to match if none of the preceding rules have matched.

The node will route a message to all outputs corresponding to matching rules. But it can also be configured to stop evaluating rules when it finds one that matches.