To install the stable version use the

Menu –

Manage palette option

Tab Install

Search for node-red-dashboard,

or run the following command in your Node-RED user directory - typically ~/.node-red:

npm i node-red-dashboard

Restart your Node-RED instance and you should have UI nodes available in the palette and a new dashboard tab in the right side panel.

The UI interface is available at <http://localhost:1880/ui> (if the default settings are used).

If you want to try the latest version from github, you can install it by

npm i node-red/node-red-dashboard

The default url for the dashboard is based off your existing Node-RED httpRoot path with /ui added. This can be changed in your Node-RED settings.js file.

ui: { path: "ui" },

You can also add your own express middleware to handle the http requests by using the ui: { middleware: your\_function } property in settings.js. For example

ui: { middleware: function (req, res, next) {

// Do something more interesting here.

console.log('LOGGED')

next()

}

},

You can also add middleware to the websocket connection using

ui: { ioMiddleware: function (socket, next) {

// Do something more interesting here.

console.log('HELLO')

next()

}

},

**Note**: both of these also accept an array of functions if you need to pass in multiple middleware actions.

Setting your own ioMiddleware will disable the default cross domain origin check.

You can also set the dashboard to be read only by ui: { readOnly: true }. This does not stop the user interacting with the dashboard but does ignore all updates coming from the dashboard.

Finally you can customise the default Group name (for i18n) by setting

ui: { defaultGroup: "Better Default" }

You can of course combine any combination of these properties

Layout

The dashboard layout should be considered as a grid.

Each **group** element has a width - by default 6 'units' (a unit is 48px wide by default with a 6px gap).

Each **widget** in the group also has a width - by default, 'auto' which means it will fill the width of the group it is in, but you can set it to a fixed number of units.

The layout algorithm of the dashboard always tries to place items as high and to the left as they can within their container - this applies to how groups are positioned on the page, as well as how widgets are positioned in a group.

Given a group with width 6, if you add six widgets, each with a width of 2, then they will be laid out in two rows - three widgets in each.

If you add two groups of width 6, as long as your browser window is wide enough, they will sit alongside each other. If you shrink the browser, at some point the second group will shift to be below the first, in a column.

It is advisable to use multiple groups if possible, rather than one big group, so that the page can dynamically resize on smaller screens.

Features

Dashboard sidebar

The widget layout is managed by a dashboard tab in the sidebar of the Node-RED editor.

Layout

* **Tabs** - From here you can re-order the tabs, groups and widgets, and add and edit their properties. You can also open the layout tools that may help you organise the widgets more easily than via the sidebar.
* **Links** - to other web pages can also be added to the menu. They can optionally be opened in an iframe - if allowed by the target page.

Site

* **Title** - the title of the UI page can be set.
* **Options** - optionally hide the title bar, and allow swiping sideways between tabs on a touch screen. You can also set whether the template uses the selected theme or uses the underlying Angular Material theme. You can also choose to use the Angular Material theme everywhere.
* **Date Format** - sets the default date format for chart and other labels.
* **Sizes** - sets the basic geometry of the grid layout in pixels. The **width** and **height** of widgets can be set, as can the width of *groups*. These are the basic definitions of the "units' used elsewhere within the dashboard.

Theme

* **Style** - the theme and font of the UI is set in the dashboard sidebar. You can select a default Light, Dark or Custom Theme. You cannot have different themes for each tab.

You can also choose to use the basic Angular Material themes instead if you like, either just within any ui\_templates or for the whole Dashboard. This will only affect angular components so some of the charts and so on may need extra work.

**Note**: For users creating their own templates the following CSS variable names are available to help pick up the theme colours.

* --nr-dashboard-pageBackgroundColor
* --nr-dashboard-pageTitlebarBackgroundColor
* --nr-dashboard-pageSidebarBackgroundColor
* --nr-dashboard-groupBackgroundColor
* --nr-dashboard-groupTextColor
* --nr-dashboard-groupBorderColor
* --nr-dashboard-widgetColor
* --nr-dashboard-widgetTextColor
* --nr-dashboard-widgetBgndColor

Widgets

Don't forget there are also extra ui widgets available on the [Node-RED flows](http://flows.nodered.org/) website. Search for node-ui- or contrib-ui- .

Group labels are optional.

Most widgets can have a label and value - both of these can be specified by properties of the incoming msg if required, and modified by angular filters. For example the label can be set to {{msg.topic}}, or the value could be set to {{value | number:1}}% to round the value to one decimal place and append a % sign.

Each node may parse the msg.payload to make it suitable for display. This converted version is exposed as the variable called value, (see example above).

Any widget can be disabled by passing in a msg.enabled property set to false;. *Note:* this doesn't stop the widget receiving messages but does stop inputs being active and does re-style the widget.

Most ui widgets can also be configured by using a msg.ui\_control message - see [**config-fields.md**](https://github.com/node-red/node-red-dashboard/blob/master/config-fields.md) for futher details.

* **Audio out** - a widget that will let you play audio (wav or mp3) or send Text to Speech (TTS) to the client.
* **Button** - the icon can be set using either Material or fa-icons - the colour and background colour may also be set. If the widget is sized to 1 wide the icon has precedence.
* **Chart** - has both line, bar and pie chart modes. Also the X-Axis labels can be customised using a date formatter string. See [**this document**](https://github.com/node-red/node-red-dashboard/blob/master/Charts.md) for more information on the chart data formats accepted.
* **Colour Picker** - a colour picker widget.
* **Date Picker** - a date picker widget. The displayed Date format can be specified in the Site tab using moment.js formatting.
* **Dropdown** - a dropdown select widget has been added. Multiple label, value pairs can be specified. The choices can also be set via msg.options containing an array of objects. If just text then the value will be the same as the label, otherwise you can specify both by using an object of "label":"value" pairs :

[ "Choice 1", "Choice 2", {"Choice 3": 3} ]

Setting msg.payload will pre-select the value in the dropdown.

* **Form** - a widget that can be composed of several sub-widgets. When submitted all values are submitted as a single message.
* **Gauge** - has 4 modes - *standard* (simple gauge), *donut* (complete 360°), *compass*, and *wave*. You can also specify the colour range of the standard and donut gauges.
* **Notification** - creates alerts to the user - can either be a toast popup, or a dismissable alert box. The alert may be targeted to a single user.
* **Numeric** - a Numeric input widget with up/down buttons.
* **Slider** - a simple horizontal slider, with variable step size.
* **Switch** - can also set two icons and/or colours depending on state.
* **Template** - the template node allows the user to specify and create their own widgets within the framework using HTML, Javascript. This is an Angular.js widget. You may also use this to override the built in CSS styles.
* **Text** - A read only widget, the layout of the label, and value can be configured.
* **Text input** - text input box, with optional label, can also support password, email and colour modes.
* **UI-Control** - allows some dynamic control of the dashboard. Sending a msg.payload of the tab number (from 0) or tab\_name will switch to that tab. Tabs can be enabled/disabled/hide/show via msg like {"tabs":{"hide":["tab\_name\_with\_underscores"],"show":["another\_tab\_name"],"disable":["unused\_tab\_name"]}}. Groups can be hidden and made visible via a msg like {"group":{"hide":["tab\_name\_group\_name\_with\_underscores"],"show":["tab\_name\_another\_group"],"focus":true}}. Outputs a msg.payload for every browser *connect* and *loss*, and every tab *change*. This can be used to trigger other actions like resetting the visibility of tabs and groups.

**Tip:** The *Text* widget will accept html - so you can use it together with the *fa-icons* we already use to create indicator type widgets.

Icons

The dashboard has 4 sets of icons built in. They are

* [Angular Material icons](https://klarsys.github.io/angular-material-icons/) : e.g. send
* [Font Awesome 4.7](https://fontawesome.com/v4.7.0/icons/) : e.g. fa-fire fa-2x
* [Weather Icons Lite](https://github.com/Paul-Reed/weather-icons-lite/blob/master/css_mappings.md) : e.g. wi-wu-sunny
* [Material Design Iconfont](https://jossef.github.io/material-design-icons-iconfont/) ; e.g. mi-alarm\_on - note add mi- to the icon name in the iconset.

And one that can only be used if you have a permanent connection to the internet

* [Iconify](https://iconify.design/icon-sets/) : e.g. iconify-mdi:car-battery 48px

Again note you have to add iconify- to the icon name in the icon set of your choice. You may also optionally specify a size in standard px or em notation. Default is 24px. You must also add a **ui\_template** node that loads the necessary iconify library into the header of the dashboard. It should contain

<script src="https://code.iconify.design/1/1.0.7/iconify.min.js"></script>

Once you have done that then you can also use them more generally, for example

<span class="iconify icon:wi:sunset icon-inline:false"></span>

You may also create your own set of icons using [Icofont](https://icofont.com/icons). Once downloaded you can serve them locally via Node-RED and add them to the head of the dashboard page by using a ui\_template node : e.g.

<link rel="stylesheet" href="myserver/path/icofont.css">

then you can use then as per above by adding the icofont- prefix e.g. icofont-badge

Or just use them in a template

<link rel="stylesheet" href="/css/icofont.css">

<div style="display: flex;height: 100%;justify-content: center;align-items: center;">

<i class="icofont icofont-4x icofont-hail"></i>

</div>

And finally via an image - https://{myserver/path/image.png}

You can use them in any of the Icon fields. You may also be able to use some of them for example in labels via their inline style eg <i class="fa fa-flag"></i>

Loading the Dashboard

Due to the size of the dashboard libraries it can take a long time to load if you are running on wireless network. It is possible add a custom loading page if you wish. To do so add a file called loading.html to the node\_modules/node-red-dashboard/dist/ folder. A simple example could be

<div><i class="fa fa-spin fa-5x fa-spinner"></i></div>

Securing the Dashboard

You can use the httpNodeAuth property in your Node-RED settings.js file to secure the Dashboard as it is created the same way as other HTTP nodes are. The details can be found at the bottom of this page in the docs <http://nodered.org/docs/security>

Multiple Users

This Dashboard does NOT support multiple individual users. It is a view of the status of the underlying Node-RED flow, which itself is single user. If the state of the flow changes then all clients will get notified of that change.

Messages coming from the dashboard **do** have a msg.socketid, and updates like change of tab, notifications, and audio alerts will be directed only to that session. Delete the msg.sessionid to send to all sessions.