

Custom dashboards

You can:

- create your own dashboards using simple HTML (no javascript is required for basic dashboards)
- utilizing any or all of the available chart libraries, on the same dashboard
- using data from one or more netdata servers, on the same dashboard
- host your dashboard HTML page on any web server, anywhere

netdata charts can also be added to existing web pages.

Check this [very simple working example of a custom dashboard](http://netdata.firehol.org/demo.html) [http://netdata.firehol.org/demo.html], and its [html source](https://github.com/netdata/netdata/tree/master/./web/gui/custom/./demo.html) [https://github.com/netdata/netdata/tree/master/./web/gui/custom/./demo.html].

You should also look at the [custom dashboard template](https://my-netdata.io/dashboard.html) [https://my-netdata.io/dashboard.html], that contains samples of all supported charts. The code is [here](https://github.com/netdata/netdata/tree/master/./web/gui/custom/./dashboard.html) [https://github.com/netdata/netdata/tree/master/./web/gui/custom/./dashboard.html].

If you plan to put the dashboard on TV, check [tv.html](#)

[https://github.com/netdata/netdata/tree/master/./web/gui/custom/./tv.html]. This is a screenshot of it, monitoring 2 servers on the same page:



Web directory

All of the mentioned examples are available on your local netdata installation (e.g. <http://myhost:19999/dashboard.html>). The default web root directory with the HTML and JS code is `/usr/share/netdata/web`. The main dashboard is also in that directory and called `index.html`.

Note: `index.html` has a different syntax. Don't use it as a template for simple custom dashboards.

Example empty dashboard

If you need to create a new dashboard on an empty page, we suggest the following header:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Your dashboard</title>

  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <meta name="apple-mobile-web-app-capable" content="yes">
  <meta name="apple-mobile-web-app-status-bar-style" content="black-translucent">

  <!-- here we will add dashboard.js -->

</head>
<body>

<!-- here we will add charts -->

</body>
</html>
```

dashboard.js

To add netdata charts to any web page (dedicated to netdata or not), you need to include the `/dashboard.js` file of a netdata server.

For example, if your netdata server listens at `http://box:19999/`, you will need to add the following to the `head` section of your web page:

```
<script type="text/javascript" src="http://box:19999/dashboard.js"></script>
```

what dashboard.js does?

`dashboard.js` will automatically load the following:

1. `dashboard.css`, required for the netdata charts
2. `jquery.min.js`, (only if jquery is not already loaded for this web page)
3. `bootstrap.min.js` (only if bootstrap is not already loaded) and `bootstrap.min.css`.

You can disable this by adding the following before loading `dashboard.js`:

```
<script>var netdataNoBootstrap = true;</script>
```

4. `jquery.nanoscroller.min.js`, required for the scrollbar of the chart legends.
5. `bootstrap-toggle.min.js` and `bootstrap-toggle.min.css`, required for the settings toggle buttons.
6. `font-awesome.min.css`, for icons.

When `dashboard.js` loads will scan the page for elements that define charts (see below) and immediately start refreshing them. Keep in mind more javascript modules may be loaded (every chart library is a different javascript file, that is loaded on first use).

Prevent dashboard.js from starting chart refreshes

If your web page is not static and you plan to add charts using javascript, you can tell `dashboard.js` not to start processing

charts immediately after loaded, by adding this fragment before loading it:

```
<script>var netdataDontStart = true;</script>
```

The above, will inform the `dashboard.js` to load everything, but not process the web page until you tell it to. You can tell it to start processing the page, by running this javascript code:

```
NETDATA.start();
```

Be careful not to call the `NETDATA.start()` multiple times. Each call to this function will spawn a new thread that will start refreshing the charts.

If, after calling `NETDATA.start()` you need to update the page (or even get your javascript code synchronized with `dashboard.js`), you can call (after you loaded `dashboard.js`):

```
NETDATA.pause(function() {  
    // ok, it is paused  
  
    // update the DOM as you wish  
  
    // and then call this to let the charts refresh:  
    NETDATA.unpause();  
});
```

The default netdata server

`dashboard.js` will attempt to auto-detect the URL of the netdata server it is loaded from, and set this server as the default netdata server for all charts.

If you need to set any other URL as the default netdata server for all charts that do not specify a netdata server, add this before loading `dashboard.js`:

```
<script type="text/javascript">var netdataServer = "http://your.netdata.server:19999";</script>
```

Adding charts

To add charts, you need to add a `div` for each of them. Each of these `div` elements accept a few `data-` attributes:

The chart unique ID

The unique ID of a chart is shown at the title of the chart of the default netdata dashboard. You can also find all the charts available at your netdata server with this URL: `http://your.netdata.server:19999/api/v1/charts` ([example \[http://netdata.firehol.org/api/v1/charts\]](http://netdata.firehol.org/api/v1/charts)).

To specify the unique id, use this:

```
<div data-netdata="unique.id"></div>
```

The above is enough for adding a chart. It most probably have the wrong visual settings though. Keep reading...

The duration of the chart

You can specify the duration of the chart (how much time of data it will show) using:

```
<div data-netdata="unique.id"
      data-after="AFTER_SECONDS"
      data-before="BEFORE_SECONDS"
></div>
```

`AFTER_SECONDS` and `BEFORE_SECONDS` are numbers representing a time-frame in seconds.

They can be either:

- **absolute** unix timestamps (in javascript terms, they are `new Date().getTime() / 1000` . Using absolute timestamps you can have a chart showing always the same time-frame.
- **relative** number of seconds to now. To show the last 10 minutes of data, `AFTER_SECONDS` must be `-600` (relative to now) and `BEFORE_SECONDS` must be `0` (meaning: now). If you want the chart to auto-refresh the current values, you need to specify **relative** values.

Chart sizes

You can set the size of the chart using this:

```
<div data-netdata="unique.id"
      data-width="WIDTH"
      data-height="HEIGHT"
></div>
```

`WIDTH` and `HEIGHT` can be anything CSS accepts for width and height (e.g. percentages, pixels, etc).

Keep in mind that for certain chart libraries, `dashboard.js` may apply an aspect ratio to these.

If you want `dashboard.js` to remember permanently (browser local storage) the dimensions of the chart (the user may resize it), you can add: `data-id="SETTINGS_ID"` , where `SETTINGS_ID` is anything that will be common for this chart across user sessions.

Netdata server

Each chart can get data from a different netdata server. You can give per chart the netdata server using:

```
<div data-netdata="unique.id"
      data-host="http://another.netdata.server:19999/"
></div>
```

If you have ephemeral monitoring setup ([More info here](#) [`../../streaming/#monitoring-ephemeral-nodes`]) and have no direct access to the nodes dashboards, you can use the following:

```
<div data-netdata="unique.id"
      data-host="http://yournetdata.server:19999/host/reported-hostname"
></div>
```

Chart library

The default chart library is `dygraph` . You set a different chart library per chart using this:

```
<div data-netdata="unique.id"
      data-chart-library="gauge"
></div>
```

Each chart library may support more chart-library specific settings. Please refer to the documentation of the chart library you are interested, in this wiki or the source code:

- options `data-dygraph-XXX` [here](#)
[<https://github.com/netdata/netdata/blob/643cfe20a8d8beba0ed31ec6afaade80853fd310/web/dashboard.js#L6251-L6361>]

- options `data-easypiechart-XXX` [here](#)
[https://github.com/netdata/netdata/blob/643cfe20a8d8beba0ed31ec6afaade80853fd310/web/dashboard.js#L7954-L7966]
- options `data-gauge-XXX` [here](#)
[https://github.com/netdata/netdata/blob/643cfe20a8d8beba0ed31ec6afaade80853fd310/web/dashboard.js#L8182-L8189]
- options `data-d3pie-XXX` [here](#)
[https://github.com/netdata/netdata/blob/643cfe20a8d8beba0ed31ec6afaade80853fd310/web/dashboard.js#L7394-L7561]
- options `data-sparkline-XXX` [here](#)
[https://github.com/netdata/netdata/blob/643cfe20a8d8beba0ed31ec6afaade80853fd310/web/dashboard.js#L5940-L5985]
- options `data-peity-XXX` [here](#)
[https://github.com/netdata/netdata/blob/643cfe20a8d8beba0ed31ec6afaade80853fd310/web/dashboard.js#L5892]

Data points

For the time-frame requested, `dashboard.js` will use the chart dimensions and the settings of the chart library to find out how many data points it can show.

For example, most line chart libraries are using 3 pixels per data point. If the chart shows 10 minutes of data (600 seconds), its update frequency is 1 second, and the chart width is 1800 pixels, then `dashboard.js` will request from the netdata server: 10 minutes of data, represented in 600 points, and the chart will be refreshed per second. If the user resizes the window so that the chart becomes 600 pixels wide, then `dashboard.js` will request the same 10 minutes of data, represented in 200 points and the chart will be refreshed once every 3 seconds.

If you need to have a fixed number of points in the data source retrieved from the netdata server, you can set:

```
<div data-netdata="unique.id"
      data-points="DATA_POINTS"
></div>
```

Where `DATA_POINTS` is the number of points you need.

You can also overwrite the pixels-per-point per chart using this:

```
<div data-netdata="unique.id"
      data-pixels-per-point="PIXELS_PER_POINT"
></div>
```

Where `PIXELS_PER_POINT` is the number of pixels each data point should occupy.

Data grouping method

Netdata supports **average** (the default), **sum** and **max** grouping methods. The grouping method is used when the netdata server is requested to return fewer points for a time-frame, compared to the number of points available.

You can give it per chart, using:

```
<div data-netdata="unique.id"
      data-method="max"
></div>
```

Changing rates

Netdata can change the rate of charts on the fly. So a charts that shows values **per second** can be turned to **per minute** (or any other, e.g. **per 10 seconds**), with this:

```
<div data-netdata="unique.id"
      data-method="average"
></div>
```

```
data-gtime="60"  
data-units="per minute"  
></div>
```

The above will provide the average rate per minute (60 seconds).

Use 60 for /minute, 3600 for /hour, 86400 for /day (provided you have that many data).

- The `data-gtime` setting does not change the units of the chart. You have to change them yourself with `data-units`.
- This works only for `data-method="average"`.
- `netdata` may aggregate multiple points to satisfy the `data-points` setting. For example, you request `per minute` but the requested number of points to be returned are not enough to report every single minute. In this case `netdata` will sum the `per second` raw data of the database to find the `per minute` for every single minute and then **average** them to find the **average per minute rate of every X minutes**. So, it works as if the data collection frequency was per minute.

Selecting dimensions

By default, `dashboard.js` will show all the dimensions of the chart.

You can select specific dimensions using this:

```
<div data-netdata="unique.id"  
data-dimensions="dimension1,dimension2,dimension3,..."  
></div>
```

`netdata` supports coma (,) or pipe (|) separated [simple patterns](#) [`../../libnetdata/simple_pattern/`] for dimensions. By default it searches for both dimension IDs and dimension NAMES. You can control the target of the match with: `data-append-options="match-ids"` or `data-append-options="match-names"`. Spaces in `data-dimensions=""` are matched in the dimension names and IDs.

Chart title

You can overwrite the title of the chart using this:

```
<div data-netdata="unique.id"  
data-title="my super chart"  
></div>
```

Chart units

You can overwrite the units of measurement of the dimensions of the chart, using this:

```
<div data-netdata="unique.id"  
data-units="words/second"  
></div>
```

Chart colors

`dashboard.js` has an internal palette of colors for the dimensions of the charts.

You can prepend colors to it (so that your will be used first) using this:

```
<div data-netdata="unique.id"  
data-colors="#AABBCC #DDEEFF ..."  
></div>
```

Extracting dimension values

`dashboard.js` can update the selected values of the chart at elements you specify. For example, let's assume we have a chart that measures the bandwidth of `eth0`, with 2 dimensions `in` and `out`. You can use this:

```
<div data-netdata="net.eth0"
      data-show-value-of-in-at="eth0_in_value"
      data-show-value-of-out-at="eth0_out_value"
></div>
```

My `eth0` interface, is receiving `` and transmitting ``.

Hiding the legend of a chart

On charts that by default have a legend managed by `dashboard.js` you can remove it, using this:

```
<div data-netdata="unique.id"
      data-legend="no"
></div>
```

API options

You can append netdata **REST API v1** [`../api/`] data options, using this:

```
<div data-netdata="unique.id"
      data-append-options="absolute,percentage"
></div>
```

A few useful options are:

- `absolute` to show all values are absolute (i.e. turn negative dimensions to positive)
- `percentage` to express the values as a percentage of the chart total (so, the values of the dimensions are added, and the sum of them if expressed as a percentage of the sum of all dimensions)
- `unaligned` to prevent netdata from aligning the charts (e.g. when requesting 60 seconds aggregation per point, netdata returns chart data aligned to `XX:XX:00` to `XX:XX:59` - similarly for hours, days, etc - the `unaligned` option disables this feature)
- `match-ids` or `match-names` is used to control what `data-dimensions=` will match.

Chart library performance

`dashboard.js` measures the performance of the chart library when it renders the charts. You can specify an element ID you want this information to be visualized, using this:

```
<div data-netdata="unique.id"
      data-dt-element-name="measurement1"
></div>
```

refreshed in `` milliseconds!

Syncing charts y-range

If you give the same `data-common-max="NAME"` to 2+ charts, then all of them will share the same max value of their y-range. If one spikes, all of them will be aligned to have the same scale. This is done for the `cpu interrupts` and `cpu softnet` charts at the dashboard and also for the `gauge` and `easypiecharts` of the netdata home page.

```
<div data-netdata="chart1"
      data-common-max="chart-group-1"
></div>
```

```
<div data-netdata="chart2"
      data-common-max="chart-group-1"
></div>
```

The same functionality exists for `data-common-min` .

Syncing chart units

netdata dashboards support auto-scaling of units. So, MB can become KB , GB , etc dynamically, based on the value to be shown.

Giving the same NAME with `data-common-units="NAME"` , 2+ charts can be forced to always have the same units.

```
<div data-netdata="chart1"
      data-common-units="chart-group-1"
></div>

<div data-netdata="chart2"
      data-common-units="chart-group-1"
></div>
```

Setting desired units

Charts can be scaled to specific units with `data-desired-units="UNITS"` . If the dashboard can convert the units to the desired one, it will do.

```
<div data-netdata="chart1"
      data-desired-units="GB"
></div>
```