**Topology Rendering Exercise**

# Introduction

This exercise requires you to take an example of a network topology structure represented in JSON format, and render it in both **graphical** and **hierarchy list** view on a web GUI. The Topology Rendering exercise should be built based on an open-source IoT solution platform called thingsboard (official site: <https://thingsboard.io>; source code: <https://github.com/thingsboard/thingsboard>) and several SNMP-capable devices.

# Interpreting the topology JSON structure:

* Each end node is represented in the following json array:

        ["**MAC address**", "**IPv6 address**", "**uplink speed**", "**downlink speed**", "**device type**", "**device status**"]

* If a node has additional children nodes connecting to it (ie, a repeater), this node will have the 7th field, which is an array of connected children For example:

        ["**MAC address**", "**IPv6 address**", "**uplink speed**", "**downlink speed**", "**device type**", "**device status**", **[Array of children nodes if any**] ]

## Device type and device status definition:

|  |  |
| --- | --- |
| **Device type ID** | **Meaning** |
| 4 | Master |
| 1 | Slave |
| 6 | Repeater |

|  |  |
| --- | --- |
| **Device status ID** | **Meaning** |
| 2 | TLS failure |
| 4 | Default status |
| 5 | Device Configured |

## Uplink and Downlink speeds

These values are represented in Hex string; please convert them to decimal values for actual speeds measured in Mbps

# Requirement

## Graphical View

* Each node in the JSON format is to be rendered as an icon depending on the “**device type**”. There are 3 “**device types”** in the attached example JSON. Please use different icon to represent different device type and make sure the MAC address of each node is displayed besides or within the icon
* Each icon is to have a variable background color depending on the “**device status**”. There are 3 “**device status**” in the attached example JSON and the background colors are to be rendered as the following:
  + Device status = 5 = **green** background
  + Device status = 4 = **yellow** background
  + Device status = 2 = **orange** background
* The relationship between each pair of nodes is to be rendered as one line connecting 2 icons together and each line must have speed subscript besides it to indicate the uplink and downlink speeds in decimal values. This one line represents both uplink and downlink speeds
* The line connecting 2 icons together is to have a variable line color depending on the “**uplink**” and “**downlink**” speeds as following:
  + Speed less than 10 Mbps = **red** line
  + Speed more than 10 Mbps but less than 30Mbps = **yellow** line
  + Speed over 30Mbps = **black** line

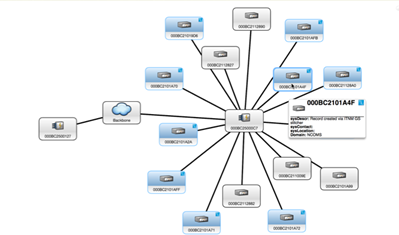
Since we are representing both uplink and downlink speeds as one line, there may have disagreement between the line colors. If disagreement occurs, render the line color according to the slower speed.

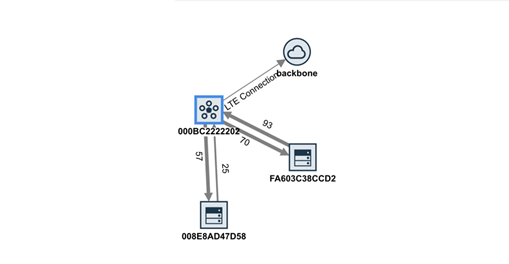
* Search Function ( considered a plus): have a simple search window that allows user to search for a particular node using the following criteria:
  + By MAC address
  + By device type
  + By device status
  + By speeds, (less than or greater than certain value)

The result of the search can be represented by one of the following way:

* + By hiding every other nodes and links that do not satisfy the search criteria, leaving only the desired ones visible.
  + By highlighting the nodes or links who satisfy the criteria
* Mouse-over display (consider a plus): when user navigates the cursor over a particular node, a small window will pop up besides the node and show additional information about this node as following:
  + MAC address
  + IPv6 address
  + Uplink speed
  + Downlink speed
  + Device type
    - Please display as human readable string according to section 2.1 rather than ID
  + Device status
    - Please display as human readable string according to section 2.1 rather than ID
* Examples:

Please note that these examples are just to show you the basic ideas, they may not look exactly as this requirement states.





## Hierarchy List View

* Represent the same topology JSON in a list view with hierarchy (connection relationships) indicated by an indentation
* Each node is represented as a line containing only MAC address
* For example:

|  |
| --- |
| * 000BC2ABCDEF   + 000BC2111111   + 000BC2222222     - 000BC2777777   + 000BC2333333   + 000BC2444444     - 000BC2555555     - 000BC2666666     - 000BC2888888 |

# Geographic Map View (bonus)

Given that the topology JSON contains one more field called “**location**” containing both **latitude and longitude** values, (for example: location = “49.24966,-123.11934”) per MAC device, please render the same topology structure on a real Geographic Map such as Google Map:

* Each MAC node is pinned on the map according to the “**location**” parameter with MAC address shown as subscript if possible.
* Connection between 2 MAC nodes is also rendered as one line according to the connection relationships defined in the JSON.