In the metrics.js file there are some events that contains function to execute serverside

```
                                if (targetNow <= 0 || (modeNow!='COOL' && modeNow != 'HEAT')) return;
                                var thejson = (modeNow=='COOL' ? { 't_cool' : --targetNow } : { 't_heat' : --targetNow });
                                exports.tstatRequest(thejson, node._id);
                            },
                        }]
                },
        //increase target temperature by 1°
        WARMER : { states: [{ label:'Warmer', action:'', icon:'fa-chevron-up', //css:'background-color:#ff1100;color:#fff',
                            serverExecute:function(node){
                                var targetNow=0, modeNow='';
                                if (node.metrics['MODE']) modeNow = node.metrics['MODE'].value;
                                if (node.metrics['TARGET']) targetNow = node.metrics['TARGET'].value;
                                if (targetNow <= 0 || (modeNow!='COOL' && modeNow != 'HEAT')) return;
                                var thejson = (modeNow=='COOL' ? { 't_cool' : ++targetNow } : { 't_heat' : ++targetNow });
                                exports.tstatRequest(thejson, node._id);
                            },
                        }],
                },
        //example presets (set specific warm/cold hold temperature in 1 click)
```

These function are passed with the node definition to the client side.

But the serverside funtions are executed in the Gateway.js using this global function “**handleNodeEvents**”

```
global.handleNodeEvents = function(node) {
  if (node.events)
  {
    for (var key in node.events)
    {
    try {
      if (node.events[key] && node.events[key].enabled)
      {
        var evt = metricsDef.events[key];
        if (evt.serverExecute!=undefined)
          try {
            evt.serverExecute(node);
          }
          catch(ex) {console.warn('Event ' + key + ' execution failed: ' + ex.message);}
      }
    } catch(ex) {console.warn('-------> EXCEPTION: nodeId:' + node._id + ' key:' + key + ' JSON: ' + JSON.stringify(node)); throw ex;}
    }
  }
}
```

Why is it not possible use a placeholder in the function’s definition for example **FunctionNameXX** and define instead the function in the gateway.js

```
                                if (targetNow <= 0 || (modeNow!='COOL' && modeNow != 'HEAT')) return;
                                var thejson = (modeNow=='COOL' ? { 't_cool' : --targetNow } : { 't_heat' : --targetNow });
                                exports.tstatRequest(thejson, node._id);
                            },
                        }]
                },
        //increase target temperature by 1°
        WARMER : { states: [{ label:'Warmer', action:'', icon:'fa-chevron-up', //css:'background-color:#ff1100;color:#fff',
                            serverExecute:function(node){
                                var targetNow=0, modeNow='';
                                if (node.metrics['MODE']) modeNow = node.metrics['MODE'].value;
                                if (node.metrics['TARGET']) targetNow = node.metrics['TARGET'].value;
                                if (targetNow <= 0 || (modeNow!='COOL' && modeNow != 'HEAT')) return;
                                var thejson = (modeNow=='COOL' ? { 't_cool' : ++targetNow } : { 't_heat' : ++targetNow });
                                exports.tstatRequest(thejson, node._id);
                            },
                        }],
                },
        //example presets (set specific warm/cold hold temperature in 1 click)
        COOL78 : { states: [{ label:'Cool:78°', action:'', icon:'fa-ge',
                            serverExecute:'FunctionNameXX'
                            },
                        }],
                },
```

gateway.js | metrics.js

```
149     }
150   }
151
152   global.sendMessageToGateway = function(msg) {
153     serial.write(msg + '\n', function () { serial.drain(); });
154   }
155
156   global.handleNodeEvents = function(node) {
157     if (node.events)
158     {
159       for (var key in node.events)
160       {
161       try {
162         if (node.events[key] && node.events[key].enabled)
163         {
164           var evt = metricsDef.events[key];
165           if (evt.serverExecute!=undefined)
166             try {
167               evt.serverExecute(node);
168             }
169             catch(ex) {console.warn('Event ' + key + ' execution failed: ' + ex.message);}
170         }
171       } catch(ex) {console.warn('-------> EXCEPTION: nodeId:' + node._id + ' key:' + key + ' JSON: ' + JSON.stringify(node)); throw ex;}
172       }
173     }
174   }
175
176
177
178
179   global.FunctionNameXX = function(node){
180     var targetNow=0, modeNow='';
181     if (node.metrics['MODE']) modeNow = node.metrics['MODE'].value;
182     if (node.metrics['TARGET']) targetNow = node.metrics['TARGET'].value;
183     if (targetNow == 78 && modeNow=='COOL') return;
184     var thejson = { 'tmode':2, 't_cool':78, 'hold':1 };
185     exports.tstatRequest(thejson, node._id);
186   }
187
188
189
190
191
192
193   //authorize handshake - make sure the request is proxied from localhost, not from the outside world
194   //if you comment out this section, you will be able to hit this socket directly at the port it's running at, from anywhere!
195   //this was tested on Socket.IO v1.2.1 and will not work on older versions
196   io.use(function(socket, next) {
197     var handshakeData = socket.request.connection;
198     console.info('AUTHORIZING CONNECTION FROM ' + handshakeData.remoteAddress + ':' + handshakeData.remotePort);
199     if (handshakeData.remoteAddress == "localhost" || handshakeData.remoteAddress == "127.0.0.1" || handshakeData.remoteAddress == "::1" ||
```

this would bring the beneficial to improve the editing of the funcion and avoid sending unnecessary information to the client side

I read something in this article that would be usefull:

https://stackoverflow.com/questions/8206453/call-function-by-string-name-on-node-js

thanks in advance

Mirko Ugolini