

FIAP

IoT - Internet das coisas (Internet of things)

Conectando dispositivos

- Agora que já exploramos as funcionalidades do Arduino e sua capacidade de conectar sensores e atuadores, vamos prosseguir conectando o Arduino a aplicações que fazem uso desse dispositivo.

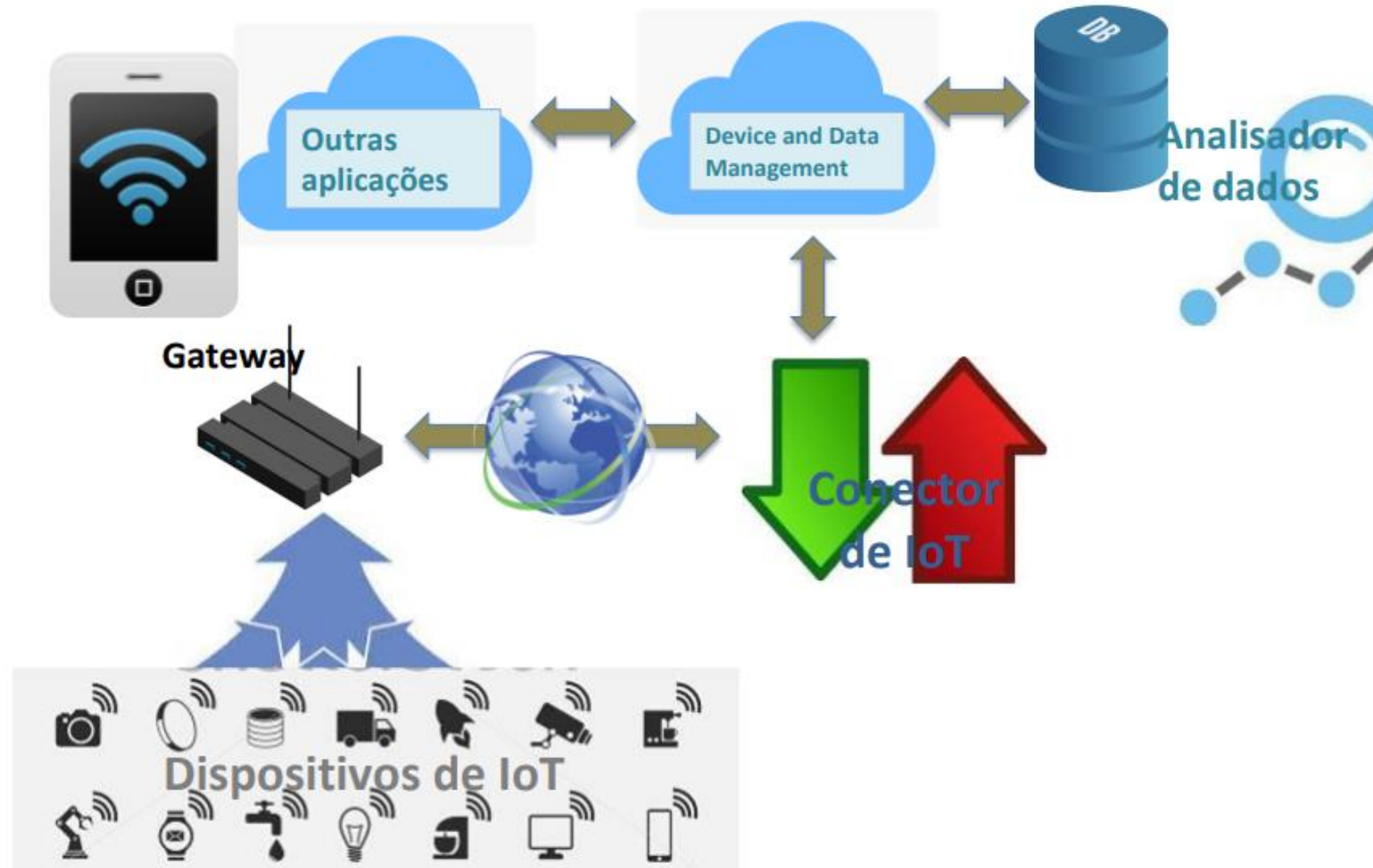
Arquitetura básica de implantação de IoT

- Arquitetura de implantação aqui fornecida é um desenho padrão para inspirar projetos reais a serem implementados, incluindo apenas os elementos fundamentais para a conectividade, sem detalhar soluções para problemas acessórios.
- IoT envolve tantas tecnologias diferentes, permitindo tantas combinações diferentes, que projetos na área tendem a se tornar “Frankensteins”

- Interoperabilidade: facilita a compatibilidade entre diferentes projetos de IoT
- Modularidade: define módulos que podem ser criados separadamente ou ainda usados “off-the-shelf” (prateleira)
- Compartilhamento de melhorias entre diferentes implementações



Arquitetura proposta



Dispositivos de IoT (devices)

- Permitem a interação com o ambiente ao seu redor, seja capturando dados de sensores como executando comandos através de seus atuadores
- Cada funcionalidade no dispositivo pode ser considerado uma Aplicação (Endpoint Application)
 - Sensores de temperatura e luminosidade são aplicações diferentes dentro da mesma placa Arduino, por exemplo
 - Cada aplicação deve ser univocamente endereçável
 - Contexto embutido em vários padrões de comunicação como USB

Conector de IoT

- Gerenciam as mensagens que chegam de dispositivos ou são destinadas a eles, adaptando-as ao protocolo de cada dispositivo
 - Em uma arquitetura de implantação pode haver conectores diferentes para protocolos diferentes
 - São capazes de identificar e autenticar dispositivos
- Apesar de poder trabalhar com HTTP, em geral fazem uso de protocolos de aplicação mais simples ou mais adequados à IoT:
 - MQTT
 - WebSocket
 - CoAP
 - LoraWan

Device and Data Management

- Faz o gerenciamento remoto dos dispositivos e de seus dados, autorizando o acesso de outras aplicações.
 - Cadastra novos dispositivos e aplicações
 - Decide se um dispositivo anunciado pode ou não ser acrescentado à rede
 - Monitora se um dispositivo está disponível no momento
- Envia comandos de gerenciamento, como:
 - Inicialização e reinicialização
 - Desligamento
 - Atualização de firmware

Banco de dados e analisador de dados

- Armazena os dados da aplicação, bem como os comandos que vão para os dispositivos
- Bancos de dados NoSQL são mais indicados, uma vez que a natureza das informações que são trocadas pelos dispositivos de IoT é muito diversa, podendo variar com o tempo
 - “Temperatura” e “Umidade”, mas tenha acabado de plugar um sensor de luminosidade...
- Faz sentido que os dados sejam monitorados por aplicações de análise de dados para um melhor aproveitamento

Gateway

- Faz a conexão de dispositivos que não tenha acesso direto à internet
 - Nem sempre é necessário
 - Quando necessário, realiza a conversão de protocolo entre os dispositivos de IoT e o conector de IoT

- O uso do IPv6 pelos dispositivos facilita a resolução do endereçamento do dispositivo, mas não é suficiente para resolver as mensagens específicas da aplicação
- Gerenciamento de múltiplos protocolos, especialmente com LAN's, PAN's e HAN's : Zigbee, Bluetooth, LoRa, Thread/6LoWPAN, etc.

Exemplo de Gateway: Philips Hue



Gateway:
Conversão entre
WiFi e Zigbee

Comunicação por Zigbee
(sem acesso ao WiFi)

- O gateway, no entanto, realiza funções simples, como autenticação do dispositivo, envio e recepção de mensagens do servidor com adaptação de protocolo, e algumas funções específicas de cada dispositivo.
- Dessa forma, a construção do gateway para o Arduino pode usar uma programação simples e visual, que explicita a origem e o destino das informações, ou seja
 - Que mensagens vêm do servidor para o dispositivo
 - Que mensagens vão do dispositivo para o servidor

Node-RED - <https://nodered.org/>

- Plataforma para programação visual de sistemas baseados em eventos
 - Executa como um servidor web na máquina hospedeira, que deve ter certas configurações mínimas de processamento e memória, como tablets e o Raspberry-Pi
 - Contribuições da comunidade – Grande disponibilidade de módulos (bibliotecas) que executam diversas funções, elaborados por empresas e desenvolvedores voluntários

Node-RED e Node.js

- O Node-Red é um serviço escrito para Node.js que provê uma ferramenta visual para editar fluxos de mensagens, vindas de diferentes fontes, podendo ser processadas e mandadas para diferentes destinos, como uma conta de email ou do Twitter.
 - A ferramenta para edição dos fluxos roda no próprio browser
 - É possível exportar e importar fluxos no formato JSON usando o menu de opções

- Uma vez que a programação do Node.js é assíncrona (não ocorre em tempo real), podemos pensar em programas em que todas as ações são acionadas por um evento gatilho.
 - Sendo assim, podemos pensar na programação do Node.js como fluxos de dados que iniciam a partir de algum evento, como o disparo de um temporizador, a requisição de um cliente de webservice ou um dado vindo do Arduino

IoT – Conhecendo Node-red

Objetivos da aula:

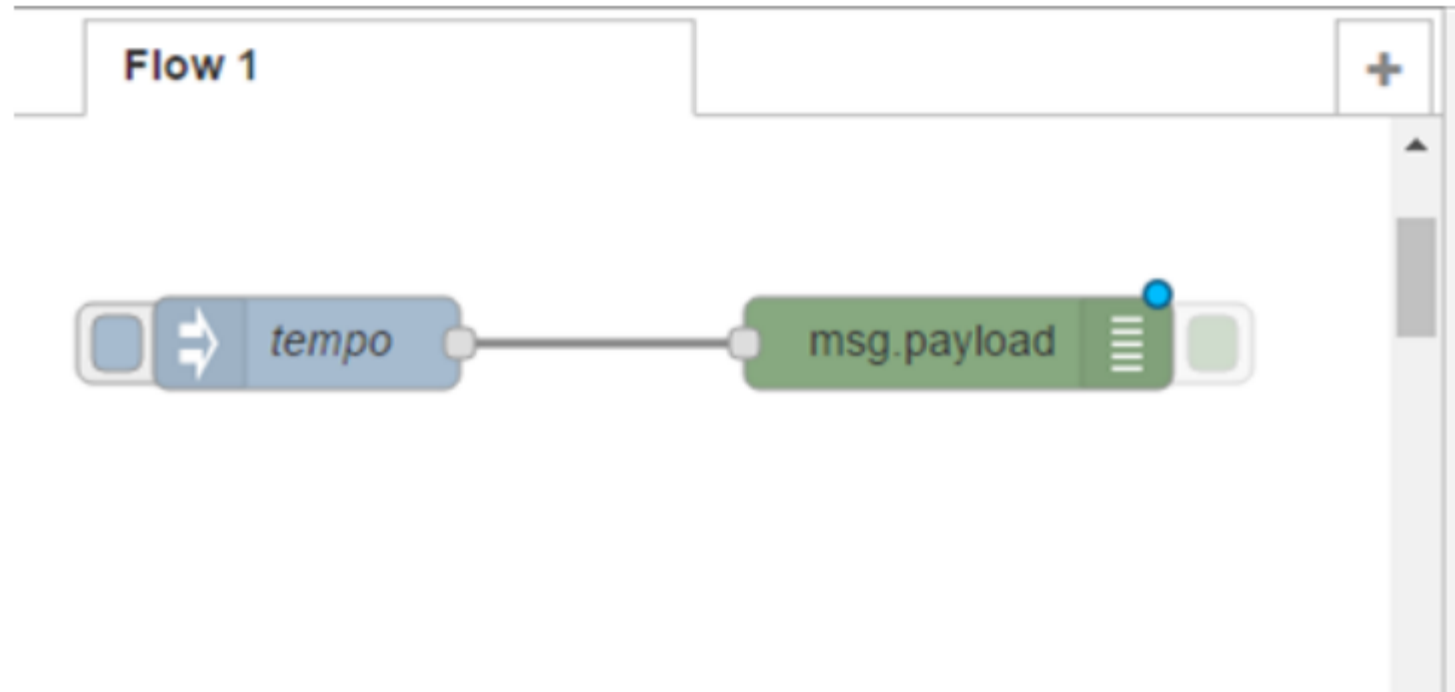
- Instalação local do Node-Red
- Primeiro fluxo (Flow) - Olá mundo!
- Dashboard - Instalar lib/Elaborar

Instalação local do Node-Red

- Faça a instalação do Node.js (versão LTS) (node –version)
 - www.nodejs.org
- Abra o cmd e digite:
`npm install -g --unsafe-perm node-red`
 - Para acessar o serviço, após instalado, digite no cmd:
`node-red`
- Acessamos no browser:
 - `http://localhost:1880`
ref: <https://nodered.org/docs/getting-started/local>

Primeiro fluxo (Flow) - Olá mundo!

- Inicialmente, ligar um nó de entrada do tipo “inject” a um nó do tipo “debug”, fazer um “Deploy” e acionar o injetor de dados
- Observar o resultado na tela de Debug
- Faça algumas alterações do inject, faça o deploy e analise o resultado



Desafio1 - Monitor do clima tempo.

- Faça o cadastro no site: Openweather <https://openweathermap.org/>
- Este site possui uma API que permite fazer requests.
- Leia a documentação: <https://openweathermap.org/current>
- Crie uma URL que faz o request do tempo em alguma cidade de sua preferência. O resultado esperado é parecido com o a imagem.

```
api.openweathermap.org/data/2.5/weather?q={city name},  
{state code},{country code}&appid={API key}
```




```

api.openweathermap.org/data/2.5/weather?q=Sao+Paulo,BR&units=metric&appid=

JSON Raw Data Headers
Save Copy Collapse All Expand All Filter JSON

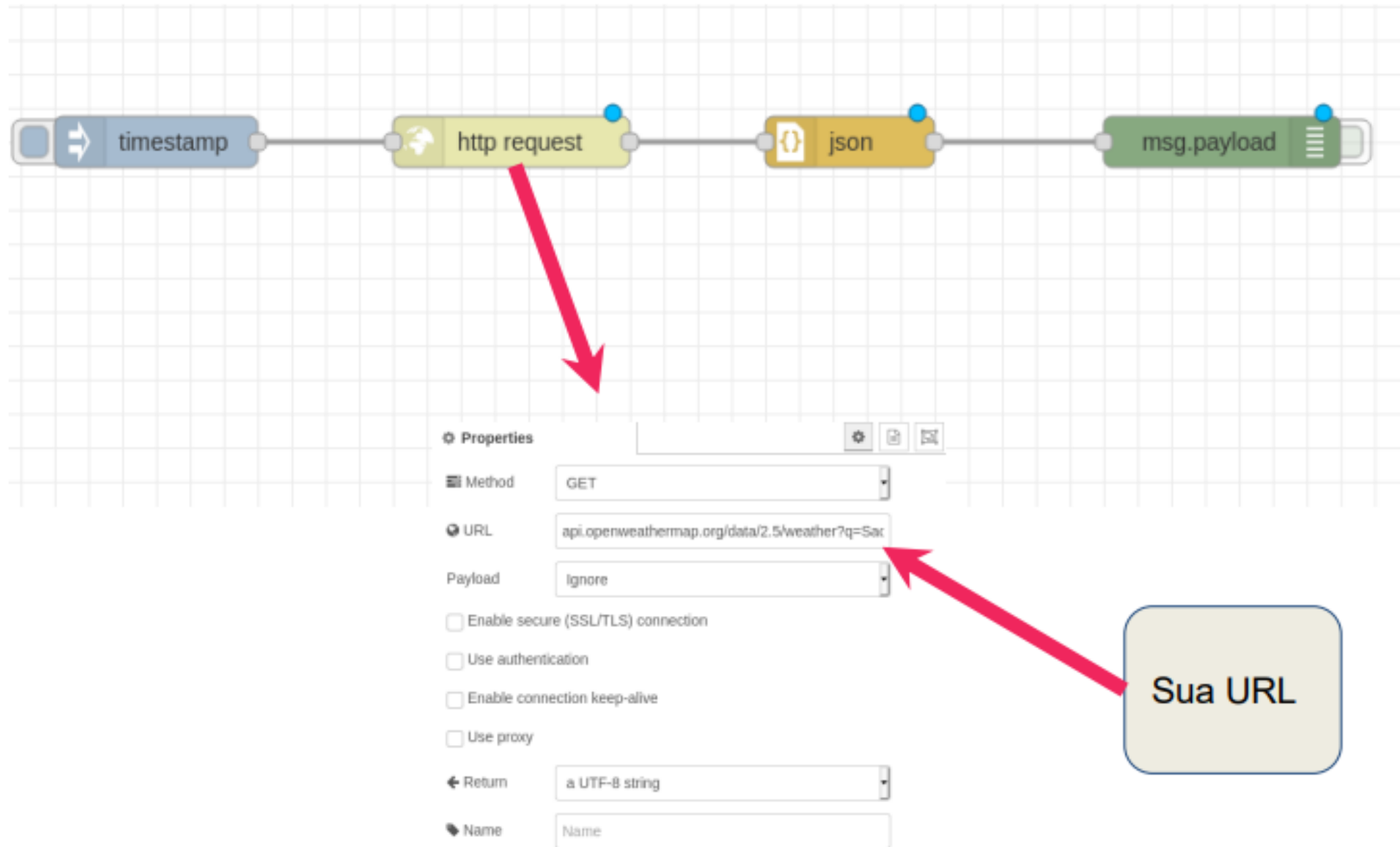
▼ coord:
  lon: -46.6361
  lat: -23.5475
▼ weather:
  0: {}
  base: "stations"
▼ main:
  temp: 15.65
  feels_like: 15.73
  temp_min: 15
  temp_max: 16.11
  pressure: 1022
  humidity: 94
  visibility: 4000
▼ wind:
  speed: 2.57
  deg: 40
▼ clouds:
  all: 75
  dt: 1620041150
▼ sys:
  type: 1
  id: 8394
  country: "BR"
  sunrise: 1620034070
  sunset: 1620074330
  timezone: -10800
  id: 3448439
  name: "São Paulo"
  cod: 200

```

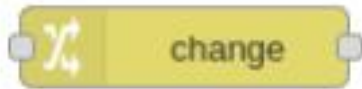
Exemplo da cidade de São Paulo.

Análise as informações disponibilizadas.

Por exemplo, onde é informada a temperatura?



- Análise o Debug do node-red e compare com o resultado obtido pelo navegador, as informações devem ser coincidentes. Algumas dessas informações não são muito relevantes e podemos filtrar.



Usando o node change, filtre apenas a “temp”



Usando o node function, escreva uma função que retorne os tópicos:
temperatura, temperatura min, temperatura max e velocidade do vento

Dica: Use o debug, e lembre a estrutura de um JSON

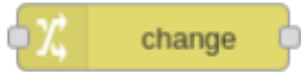
Desafio1 - Solução

Sugestão: Está na documentação oficial!

SEMPRE LEIA A DOCUMENTAÇÃO OFICIAL!

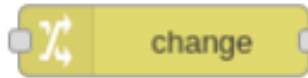
<https://nodered.org/docs/user-guide>

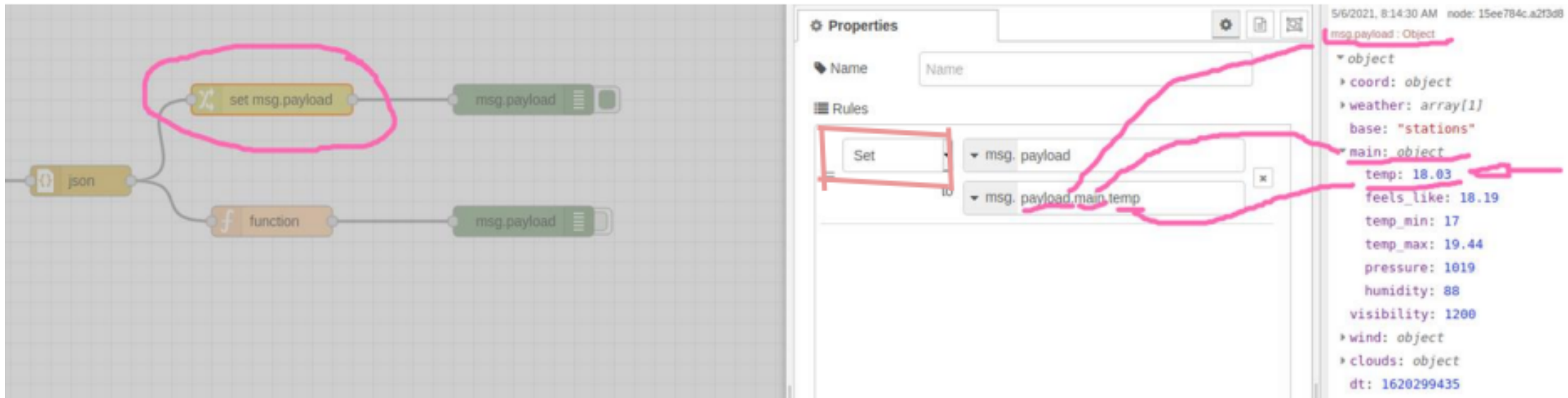
Desafio1 - Solução



- Realiza operações de modificações nos campos da mensagem, podendo modificar, acrescentar, apagar ou renomear um campo
 - Serve também para seus subcampos
 - Podem ser realizadas diversas operações consecutivas
- As operações que podem ser executadas são:
 - **Set**: cria ou modifica o valor de um campo
 - **Modify**: substitui o valor de um campo por outro
 - **Move**: renomeia um campo
 - **Delete**: apaga e remove um campo

Desafio1 - Solução

 **change** Usando o node change, filtre apenas a “temp”



O **msg.payload** carrega todas a informações que vem da API, queremos apenas a temperatura.

Vamos analisar a estrutura do msg.payload para configurar corretamente.

O resultado fica: payload.main.temp

Desafio1 - Solução

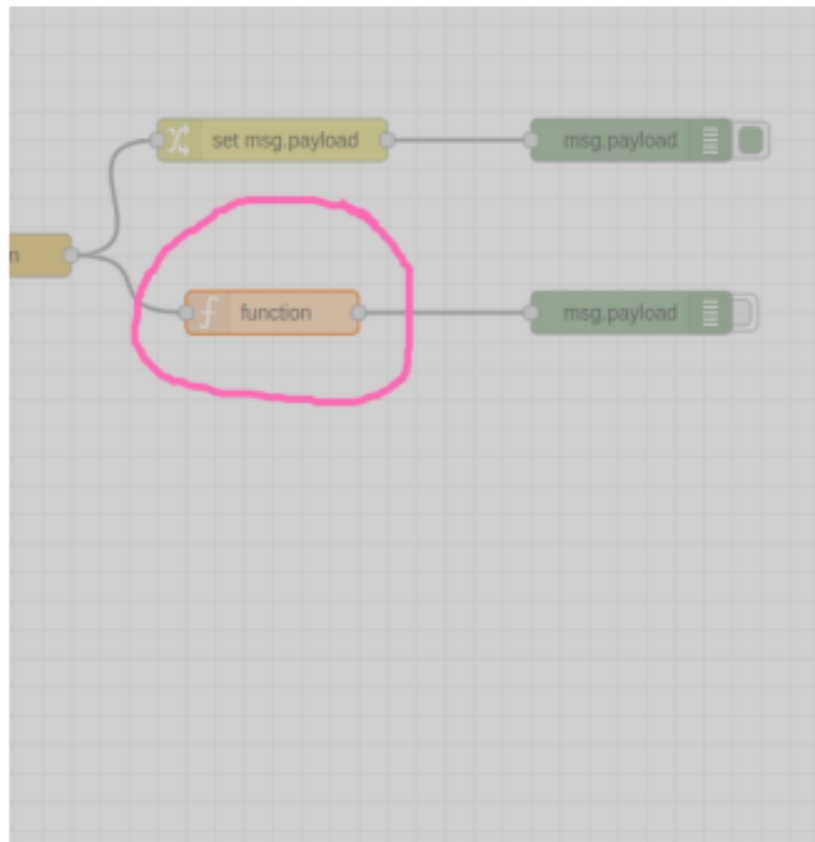


- Cria uma função JavaScript genéricas que podem manipular os campos da mensagem como se desejar.
 - A função do Javascript recebe o parâmetro msg, e pode retornar um ou mais objetos
 - Para o bom funcionamento do programa, é preciso que o(s) valor(es) retornados sejam a própria msg com as devidas modificações, ou null para interromper o caminho da mensagem
- O node Function implementa apenas o corpo da função.

Desafio1 - Solução



Usando o node function, escreva uma função que retorne os tópicos: temperatura, temperatura min, temperatura max e velocidade do vento



```
Properties
Name
Setup On Start On Message On Stop
1
2 return msg;
```

função

retorno da função

```
5/6/2021, 8:14:30 AM node: 15ee784c
msg.payload : Object
object
  coord: object
  weather: array[1]
    base: "stations"
  main: object
    temp: 18.03
    feels_like: 18.19
    temp_min: 17
    temp_max: 19.44
    pressure: 1019
    humidity: 88
    visibility: 1200
  wind: object
  clouds: object
    dt: 1620299435
  sys: object
    timezone: -10800
    id: 3448439
    name: "São Paulo"
    cod: 280
5/6/2021, 8:22:44 AM node: 15ee784c
msg.payload : number
18.03
```

Desafio1 - Solução

Note que recebemos as informações em `msg.payload` e que damos return em `msg`. Queremos as informações:

```
temp
temp_min
temp_max
speed
```

Temos algumas soluções possíveis. Uma delas, vamos criar uma saída `msg` que vai exibir os dados filtrados mantendo o formato de um objeto Java Script ("chave":valor):

```
msg.payload = {
  "chave": valor,
  "chave": valor,
  "chave": valor,
  "chave": valor }
```

return msg;

template

```
msg.payload = {
  "temperatura": msg.payload.main.temp,
  "min": msg.payload.main.temp_min,
  "max": msg.payload.main.temp_max,
  "vento": msg.payload.wind.speed
}
```

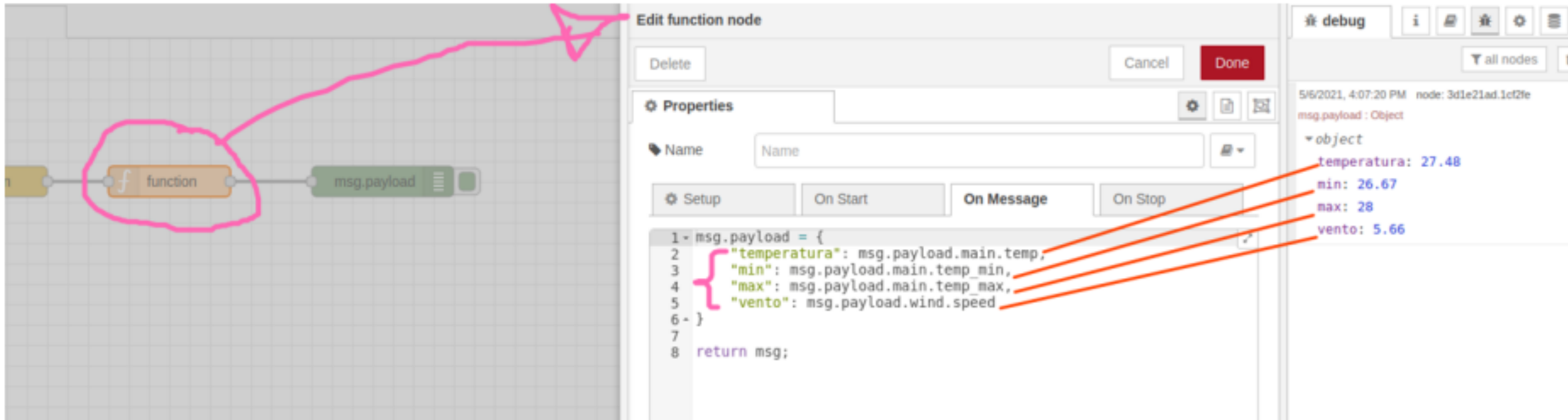
return msg;

Resultado

Desafio1 - Solução



Usando o node function, escreva uma função que retorne os tópicos: temperatura, temperatura min, temperatura max e velocidade do vento



The screenshot shows the Node-RED interface. On the left, a workflow is visible with a yellow input node connected to an orange 'function' node, which is then connected to a green 'msg.payload' output node. The 'function' node is circled in pink, and a pink arrow points from it to the 'Edit function node' dialog box on the right.

The 'Edit function node' dialog box has the following sections:

- Properties:** A text field for 'Name'.
- Setup:** A tab labeled 'On Message' is selected.
- Code:** A text area containing the following JavaScript code:

```
1- msg.payload = {  
2-   "temperatura": msg.payload.main.temp,  
3-   "min": msg.payload.main.temp_min,  
4-   "max": msg.payload.main.temp_max,  
5-   "vento": msg.payload.wind.speed  
6- }  
7-  
8- return msg;
```

Four orange arrows point from the code to the debug console on the right:

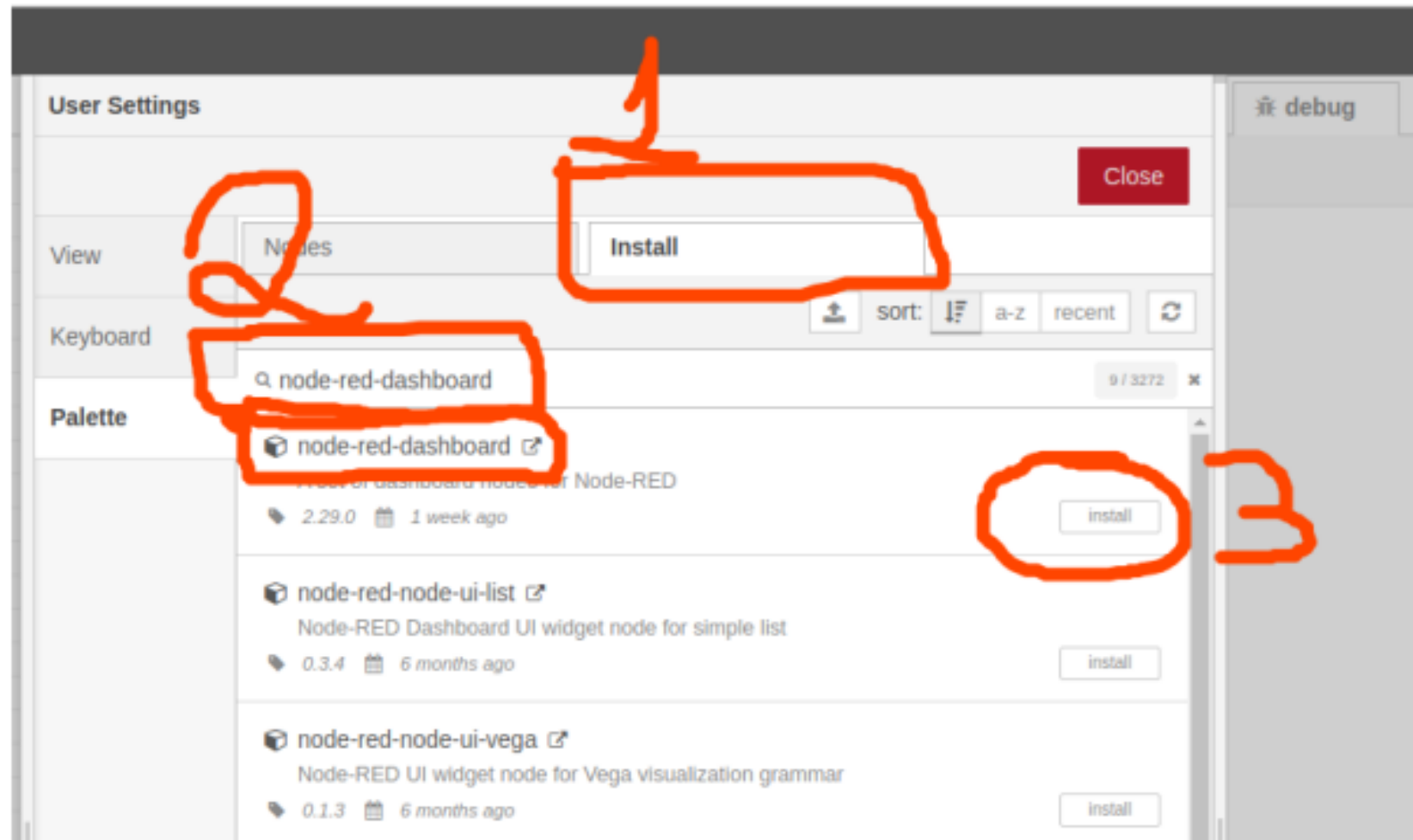
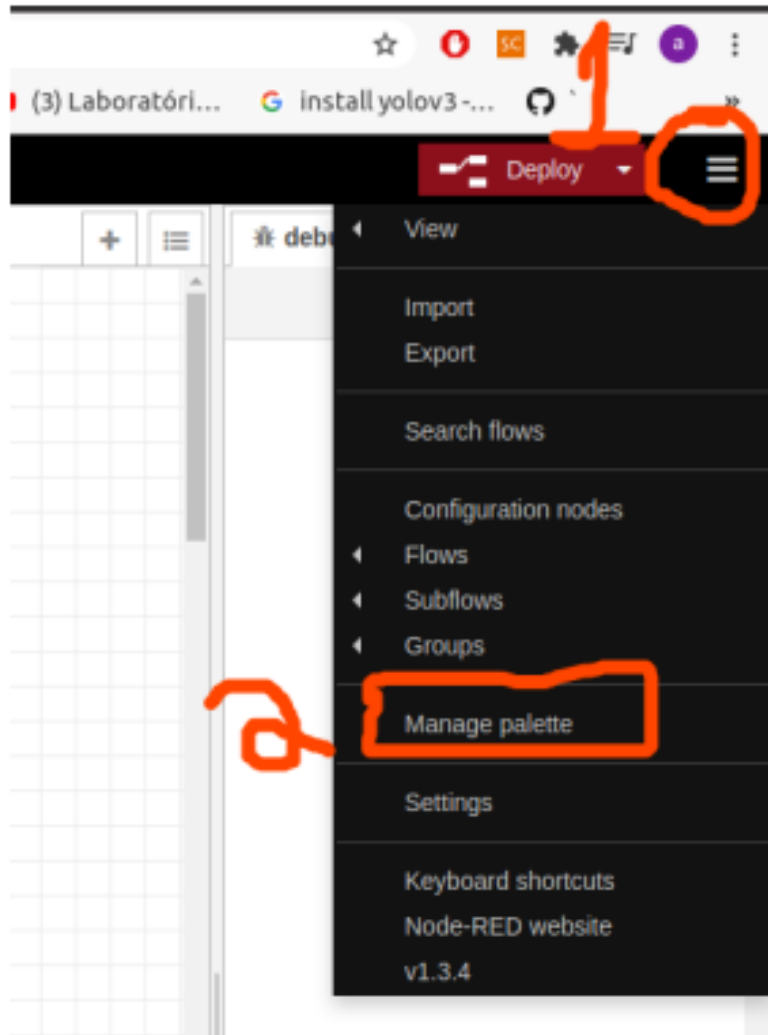
- From line 2 to 'temperatura: 27.48'
- From line 3 to 'min: 26.67'
- From line 4 to 'max: 28'
- From line 5 to 'vento: 5.66'

The debug console on the right shows the following output:

```
5/6/2021, 4:07:20 PM node: 3d1e21ad.1cf2fe  
msg.payload : Object  
  object  
    temperatura: 27.48  
    min: 26.67  
    max: 28  
    vento: 5.66
```

Dashboard - Instalar lib

- Primeira coisa, vamos instalar os nodes para dashboard, caso não já não tenha instalado.



node-red-dashboard

The image shows a Node-RED interface with a warning dialog box open. The dialog box has the title "Installing 'node-red-dashboard'" and the text: "Before installing, please read the node's documentation. Some nodes have dependencies that cannot be automatically resolved and can require a restart of Node-RED." At the bottom of the dialog are three buttons: "Cancel", "Open node information", and "Install". The "Open node information" and "Install" buttons are circled in orange. A large orange bracket is drawn around the "Open node information" button and the "node-red-dashboard" entry in the node palette below. In the node palette, the "node-red-dashboard" node is selected, showing its description: "A set of dashboard nodes for Node-RED", version "2.29.0", and "1 week ago". An "install" button is visible next to it. Below this, the "node-red-node-ui-list" node is also visible. A large orange text overlay at the bottom reads "LEIA A DOCUMENTAÇÃO".

Installing 'node-red-dashboard'

Before installing, please read the node's documentation. Some nodes have dependencies that cannot be automatically resolved and can require a restart of Node-RED.

Cancel Open node information Install

node-red-dashboard

A set of dashboard nodes for Node-RED

2.29.0 1 week ago

node-red-node-ui-list

Node-RED Dashboard UI widget node for simple list

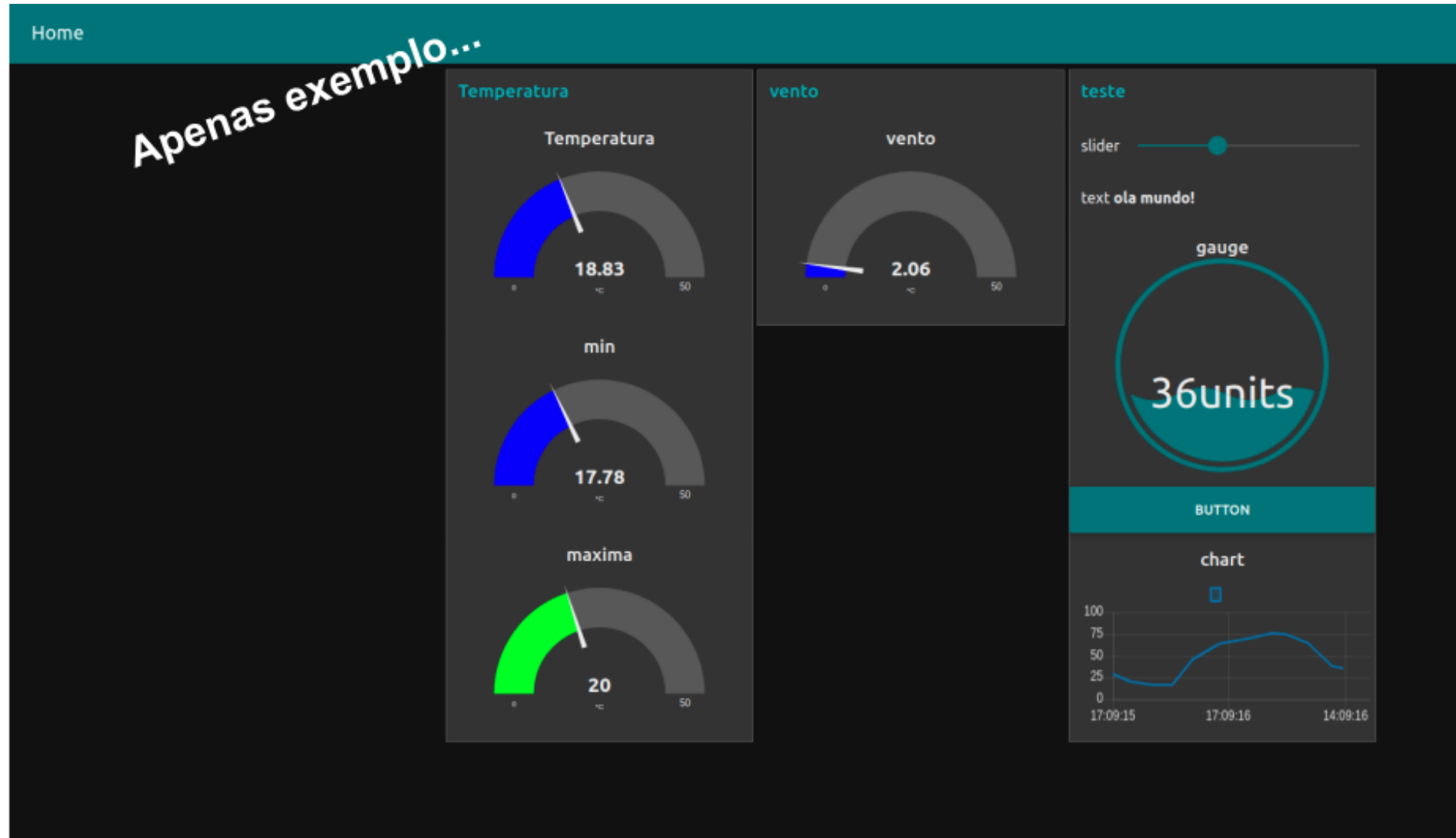
0.3.4 6 months ago

LEIA A DOCUMENTAÇÃO



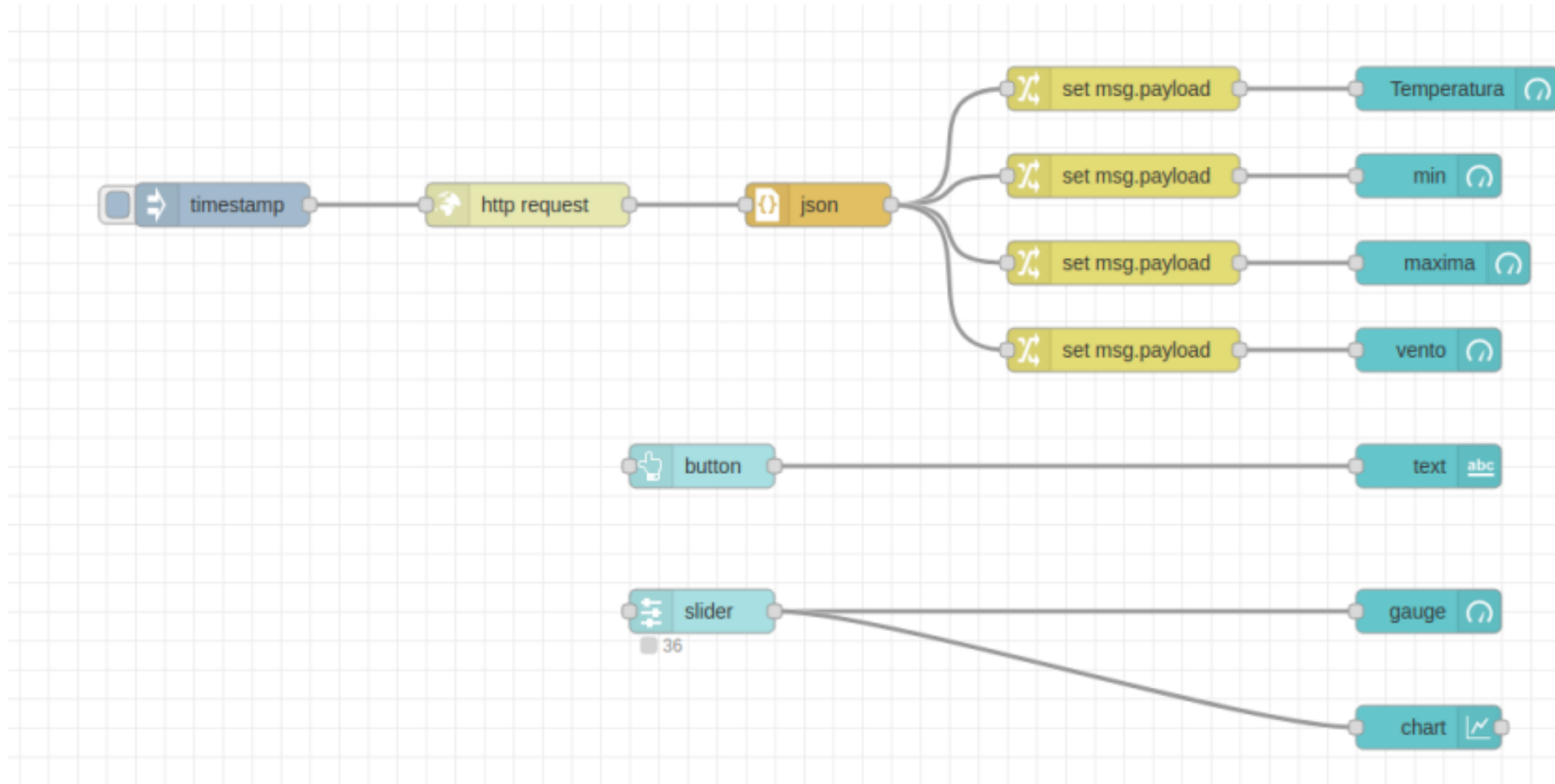
Show! Vamos usar

Dashboard - Como elaborar...



Dashboard - Como elaborar...

Monte o flow...



Desafio 2

- O dashboard de exemplo do professor está muito simples...mas é um bom começo para criar novos dashboards.
- Edite/Crie de forma criativa um dashboard para exibir as informações da API Open Weather Map.
- O mínimo esperado: Fazer a busca de pelo menos duas cidades diferentes da sua preferência, exibindo pelo menos as informações:
 - Temperatura atual;
 - Temperatura mínima;
 - Temperatura máxima;
 - Velocidade do vento;
 - Humidade relativa;
 - Sensação térmica.



**Copyright © 2023 Prof. Arnaldo Jr/Yan
Coelho**

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).