



PSI3541 2023

SISTEMAS EMBARCADOS DISTRIBUIDOS

AULA 05 14/04/2023 Introdução ao NODE-RED
PROF. SERGIO TAKEO KOFUJI - KOFUJI@USP.BR

O que é o Node-RED

- Ferramenta para criação **NO-CODE** de aplicações **IoT**
- Simplifica o desenvolvimento “ligando” blocos de código para executar tarefas.
- Faz uso de uma abordagem de **programação visual** que permite aos desenvolvedores conectar blocos de códigos predefinidos, conhecidos como "**nós**", para executar uma determinada tarefa.
- Os nós conectados, geralmente uma combinação de nós de entrada, nós de processamento e nós de saída, quando ligados juntos, formam um "**fluxo**".

Características

- É opensource –
 - <https://github.com/node-red/>
 - <https://nodered.org>
- Instalação simples
- Vasta documentação
 - Tutoriais: <https://cookbook.nodered.org/>
 - Guia de usuário: <https://nodered.org/docs/user-guide/>
 - Tutorial para criação de nós: <https://nodered.org/docs/creating-nodes/>
 - Documentação da API: <https://nodered.org/docs/api/>
 - Tutorial para desenvolvedores colaboradores: <https://nodered.org/docs/developing/>

Características

- Muitos exemplos de fluxos prontos:
 - <https://flows.nodered.org/?type=flow>
- Ampla variedade de nós (<https://flows.nodered.org/?sort=downloads&type=node>):
- Redes sociais / mensageiros: Twitter, Telegram
- Integração com outros serviços: IBM Watson, Azure, Openwhisk
- Banco de dados: MySQL, MongoDB, InfluxDB

NODE-RED USER INTERFACE

The screenshot displays the Node-RED web interface. At the top left, the Node-RED logo and name are visible. The main workspace is titled "Monitoramento de pacientes" and contains two flows. The first flow consists of a "Blood pressure Simulator" node connected to a "json" node, which is connected to a "Schema converter" node, which is finally connected to a "Blood Pressure C" node. The second flow consists of a "Glucose Simulator" node connected to a "json" node, which is connected to a "Schema converter" node, which is finally connected to a "Glucose Class" node. On the left side, there is a "Paleta de nós" (node palette) with a search bar and a list of input nodes including inject, catch, status, link, mqtt, http, websocket, tcp, and udp. At the top right, there is a "Menu de deploy" with a "Deploy" button and a hamburger menu icon. On the right side, there is an "Informações e debug" panel with a "Menu de aplicação" (application menu) containing icons for info, help, close, and a chart. The panel displays information about the current flow, including the project name "Teste", the flow ID, the flow name "Monitoramento de pacientes crônicos", and the status "Enabled". It also shows a "Flow Description" section which is currently empty. At the bottom of the panel, there is a message: "You can confirm your changes in the node edit tray with `ctrl-enter` or cancel them with `ctrl-escape`".

Paleta de nós

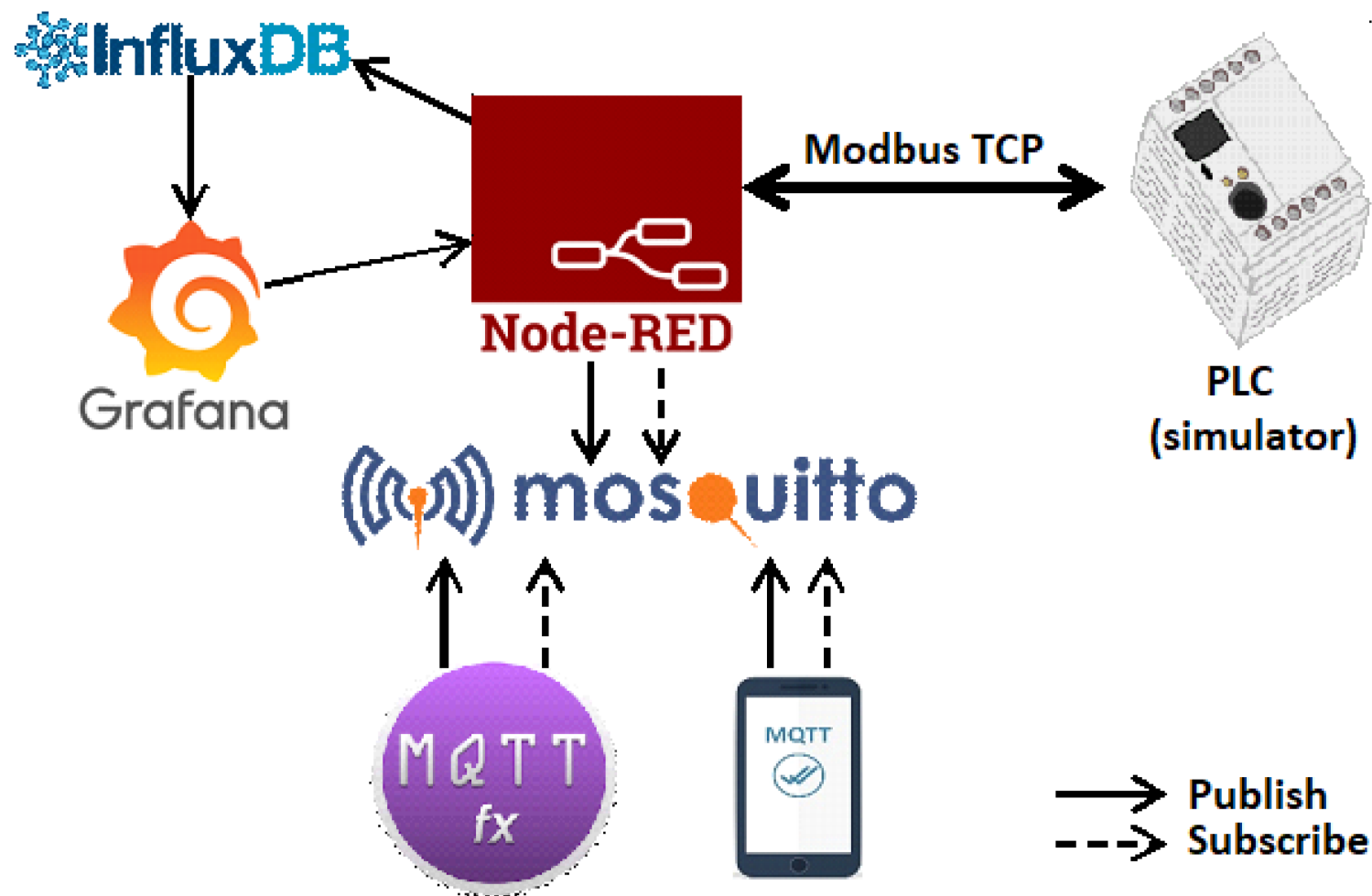
Editor de fluxo

Menu de deploy

Menu de aplicação

Informações e debug

Exemplo de uma aplicação Node-Red



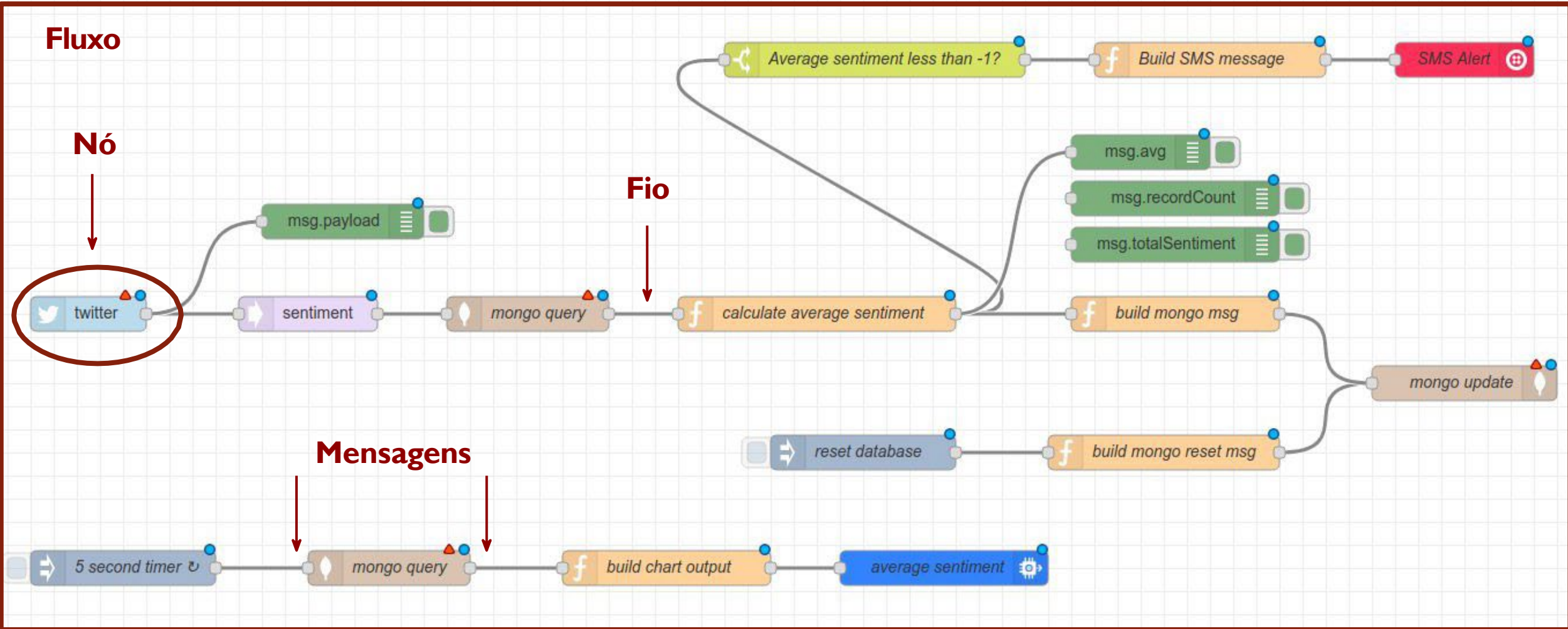
Elementos de um Fluxo Node-Red

Fluxo

Nó

Fio

Mensagens



Fluxos

- No Node-RED, programas são **fluxos** compostos por uma coleção de **nós** conectados para trocar mensagens.
- Tecnicamente, um fluxo consiste em uma lista de objetos **JavaScript** que descrevem os nós e suas configurações



Mensagens

- As mensagens transmitidas entre nós no Node-RED são, por convenção, objetos JavaScript chamados `msg`, consistindo em um conjunto de propriedades nomeadas
- Geralmente contêm uma propriedade `msg.payload` com a carga útil (payload) da mensagem.
- Os nós podem anexar outras propriedades a uma mensagem, que pode ser usada para transportar outras informações para o próximo nó no fluxo, como por exemplo o `msg.topic`

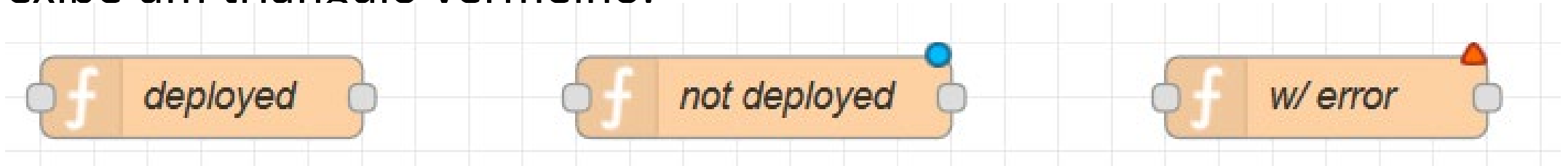
Nós

- São o principal bloco de construção dos fluxos no Node-RED.
- Quando um fluxo está em execução, as mensagens são geradas, consumidas e processadas pelos nós.
- Um nó pode ser vinculado a (múltiplas) entradas e (múltiplas) saídas por meio de suas portas, que permitem que as mensagens sejam transmitidas entre os nós.



Nós

- Se um nó foi alterado após a implantação mais recente, ele exibe um círculo azul acima dele. Se houver erros com sua configuração, ele exibe um triângulo vermelho.

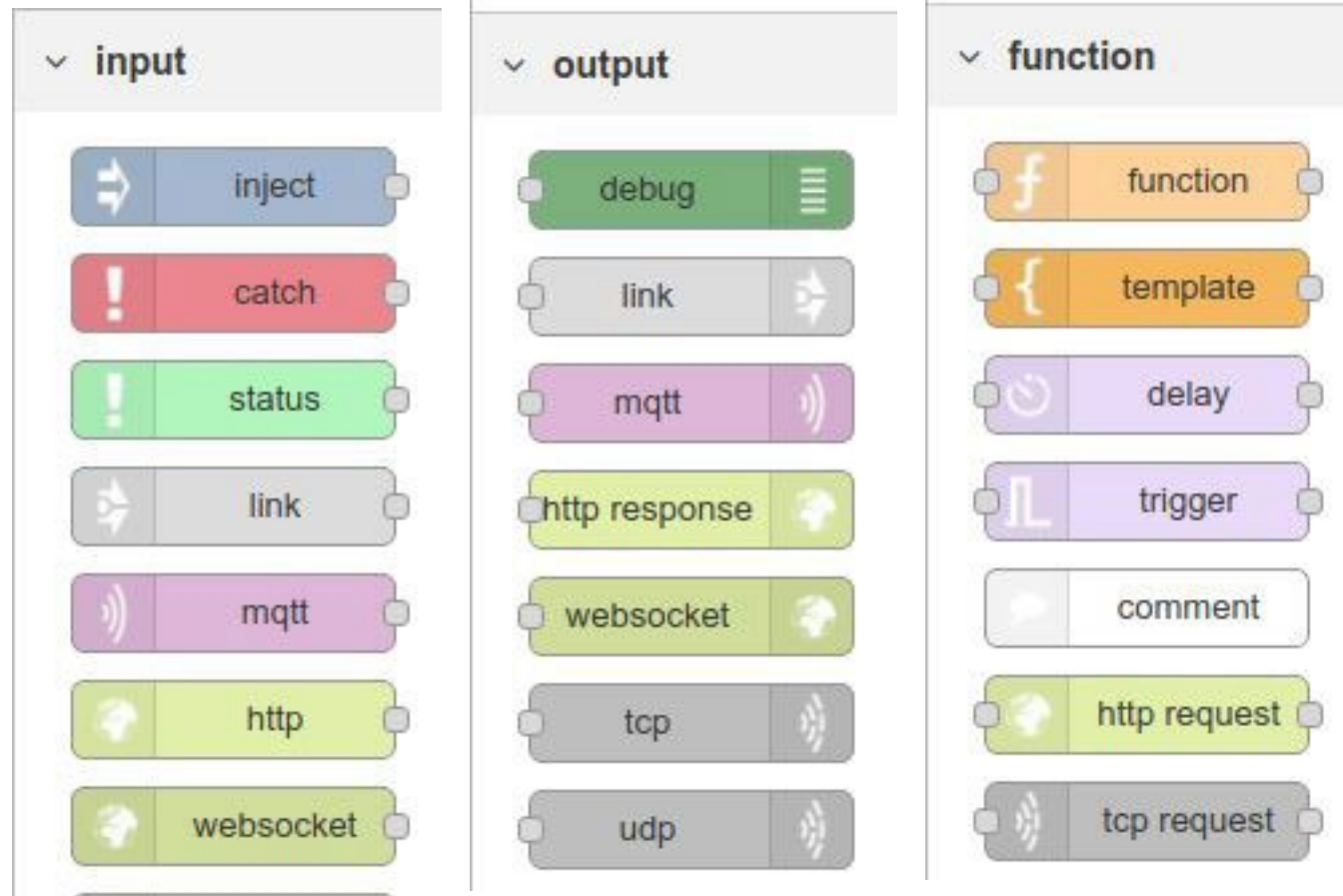


- Alguns nós incluem um botão em sua borda esquerda ou direita. Isso permite alguma interação com o nó de dentro do editor. Os nós Injetar e Depurar são os únicos nós principais que possuem botões.



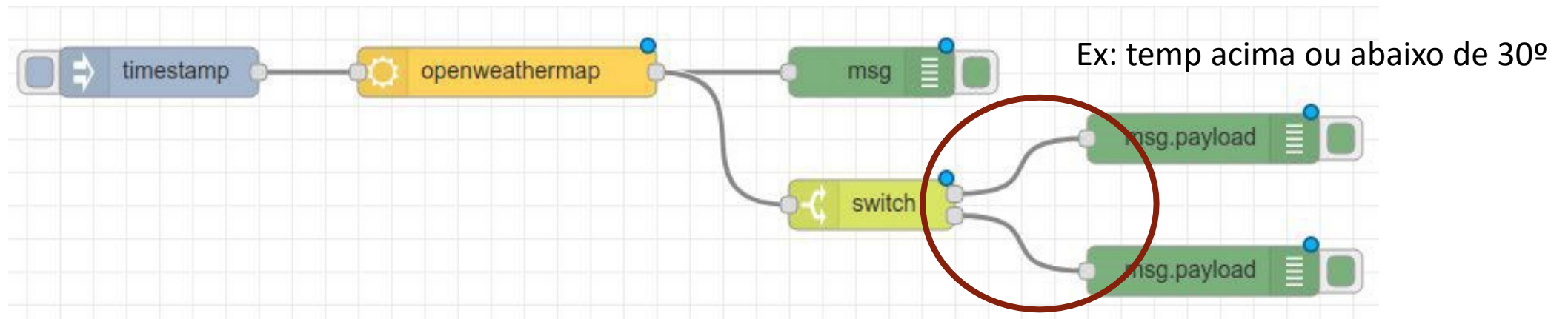
Tipos de Nós

- 3 tipos básicos
- Nós de entrada - gera mensagens p/ nós de recebimento de dados.
- Nós de saída - consome mensagens, por exemplo, para enviar dados para um serviço externo
- Nós de processamento - nós que processam dados de alguma forma, emitindo mensagens novas ou modificadas.
- 2 categorias adicionais
- Nós de credenciais
- Nós criados por usuários



Fios

- Os fios definem as conexões entre os nós de entrada e saída em um fluxo.
- Conectam a extremidade dos nós de saída às entradas dos nós de recebimento de dados.
- É possível conectar mais de um nó à extremidade do nós de saída usando fios.
- Quando vários nós são conectados a um terminal de saída, as mensagens são enviadas para cada nó conectado, na ordem em que foram conectadas à saída.



Contexto

- As mensagens, até agora, eram a única maneira de obter dados dentro e fora dos nós. Todavia, há uma exceção a essa regra -> **context**
- O **context** está disponível apenas para os nós de função
- É usado para armazenar dados na memória que podem ser acessados posteriormente
- É importante para os nós que precisam manter um índice, contar ou somar dados nas mensagens.
- Tipos de contexto
 - Contexto local (**context**) - que pode ser acessado apenas pelo nó que criou o índice
 - Contexto do fluxo (**flow**) - que pode ser acessado por todos os nós de um fluxo
 - Contexto global (**global**) - que pode ser acessado por todos os nós de uma instância

Nós de Função

- Os nós de função são o “canivete suíço” do Node-RED
- Podem ser utilizados quando não há um nó dedicado especificamente à tarefa que se deseja executar
- São escritos em Javascript por meio de um editor de código disponível no Node-RED



Edit function node

Delete Cancel Done

node properties

Name
Converts to uppercase

Function

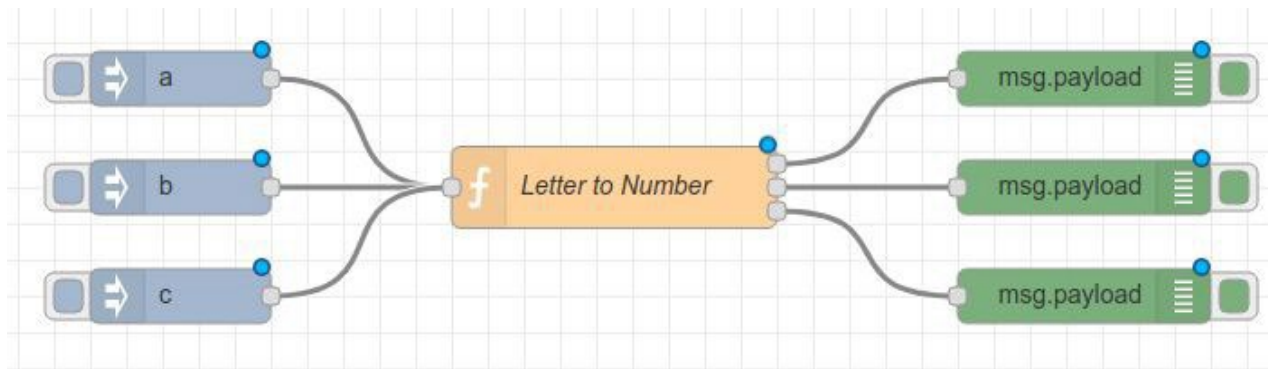
```
1 msg.payload = msg.payload.toUpperCase();  
2 return msg;
```

Outputs 1

See the Info tab for help writing functions.

Nós de função

- Múltiplos Valores de Saída



Edit function node

Delete Cancel Done

node properties

Name
Letter to Number

Function

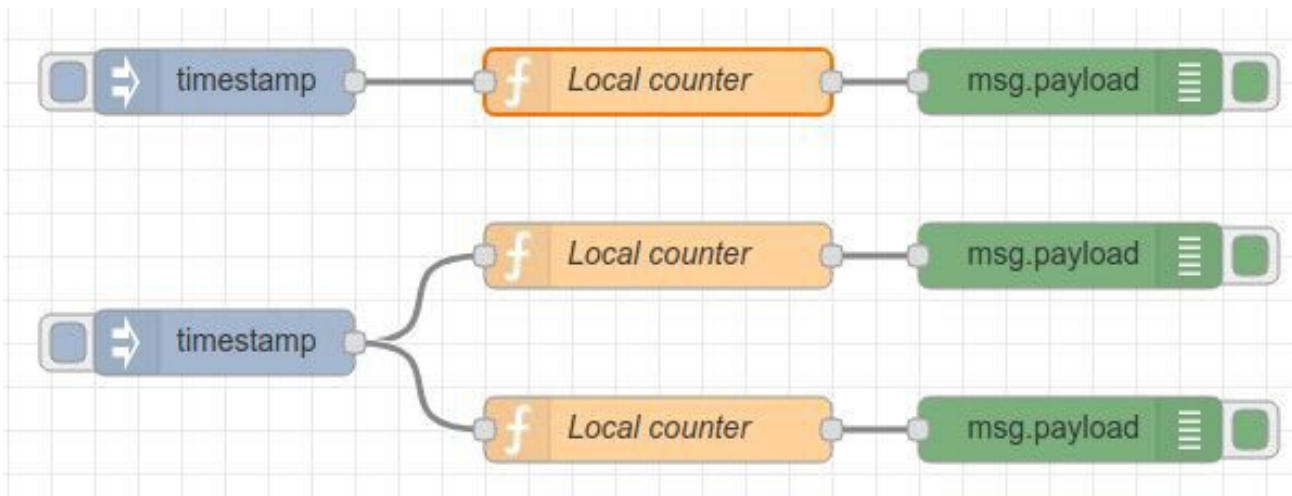
```
1 if (msg.payload == "a") {  
2   msg.payload = 1;  
3   return [ msg, null, null ];  
4 } else if (msg.payload == "b") {  
5   msg.payload = 2;  
6   return [ null, msg, null ];  
7 } else {  
8   msg.payload = 3;  
9   return [null, null, msg];  
10 }
```

Outputs 3

See the Info tab for help writing functions.

Nós de Função

- utilizando o contexto local (context)



Edit function node

Delete Cancel Done

node properties

Name
Local counter

Function

```
1 var count = context.get('count') || 0;  
2 count += 1;  
3 context.set('count', count);  
4 msg.payload = count;  
5 return msg;
```

Outputs 1

debug

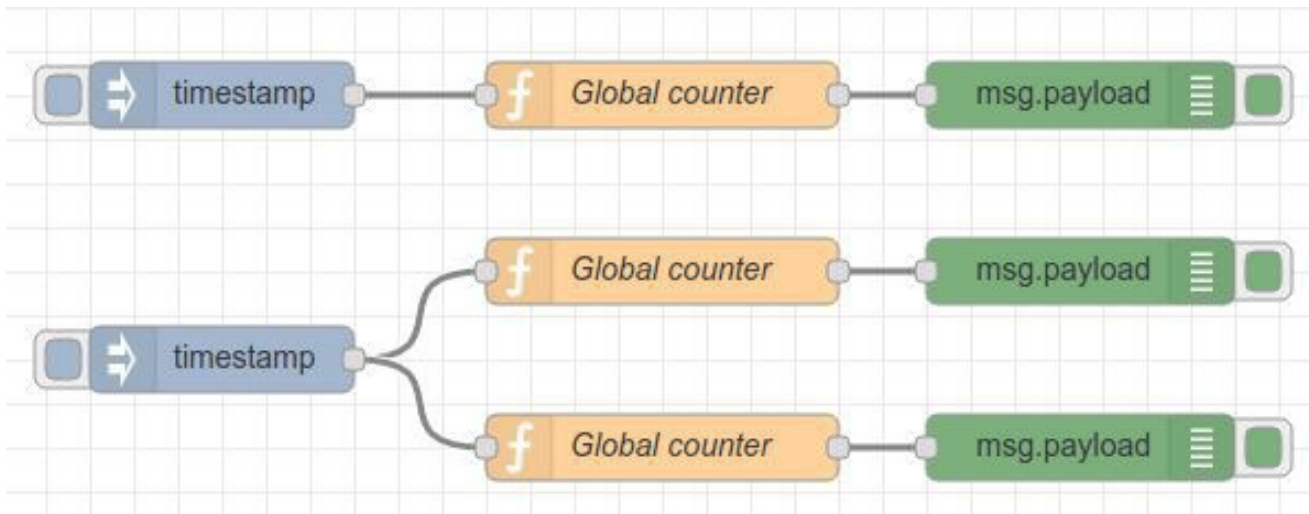
5/28/2019, 2:30:14 PM node: fc41af9c.8ab46
msg.payload : number
1

5/28/2019, 2:30:16 PM node: be4b213b.ee46b
msg.payload : number
1

5/28/2019, 2:30:16 PM node: a74ce89e.ed3ca8
msg.payload : number
1

Nós de Função

- utilizando o contexto global (global)



Edit function node

Delete Cancel Done

node properties

Name: Global counter

Function:

```
1 var count = global.get('count') || 0;  
2 count += 1;  
3 global.set('count', count);  
4 msg.payload = count;  
5 return msg;
```

Outputs: 1

debug

5/28/2019, 2:55:36 PM node: 11143079.ff95f
msg.payload : number
1

5/28/2019, 2:55:37 PM node: 78d447b1.84aa18
msg.payload : number
2

5/28/2019, 2:55:37 PM node: 6b703688.fc2d18
msg.payload : number
3

JSON

- JSON -> JavaScript Object Notation
- Utilizado para representar um objeto JavaScript como String.
- É comumente usado por APIs da web para retornar dados.
- Elementos básicos do JSON.
 - { e } - delimita um objeto.
 - [e] - delimita um array.
 - : - separa chaves (atributos) de valores.
 - , - separa os atributos chave/valor.

JSON

- Os tipos de dados básicos do JSON são:
 - string - separados por aspas (duplas ou simples). Ex. "Brasil" ou 'Brasil'
 - número - sem aspas e pode ser inteiro ou real. Ex. 1 (inteiro) ou 23.454 (real)
 - booleano - tipo lógico normal, pode assumir valores true ou false.
 - nulo - valor para representar nulo. Ex. { "nome" : null }
 - object: É um conjunto de pares nome/valor.
 - array: utilizados para elementos ordenados.

JSON

- Usualmente as mensagens trocadas pelos nós seguem o formato JSON
- Se o JSON estiver representado como String é necessário realizar uma conversão antes de utilizá-lo.
- O Node-RED fornece um nó para converter string -> json -> string

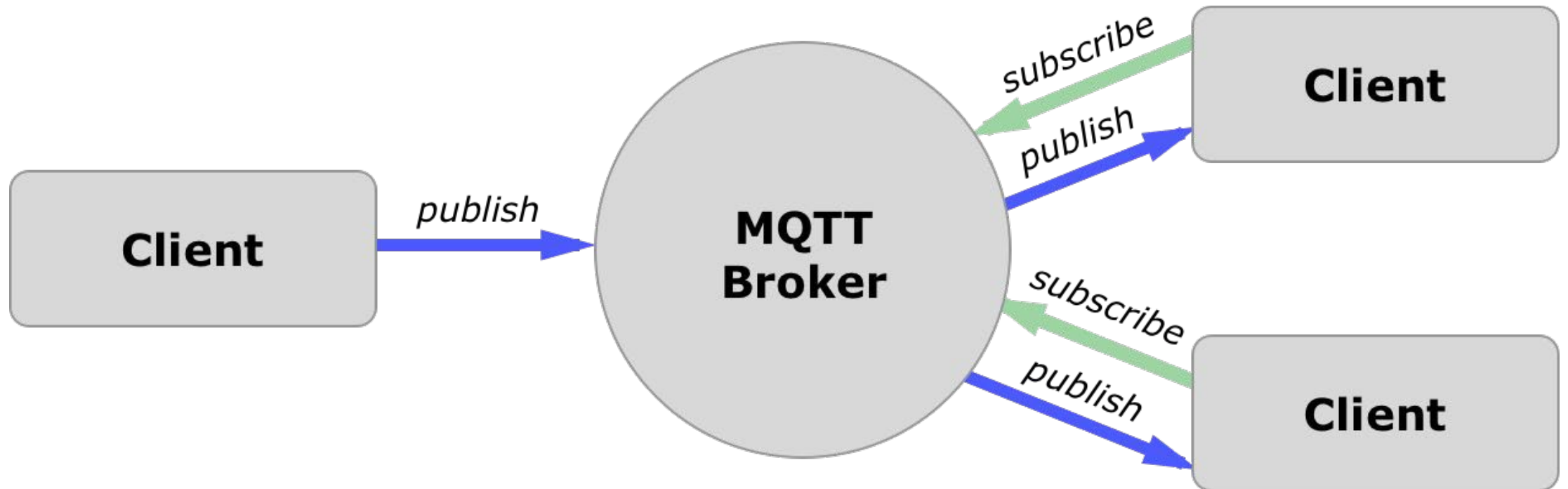
MQTT

- É um protocolo comunicação Machine-to-Machine (M2M), ou máquina a máquina
- Inventado e desenvolvido inicialmente pela IBM no final dos anos 90
- Era utilizado para conectar sensores em pipelines de petróleo a satélites
- Permite a comunicação entre pares de maneira assíncrona
- Por ser leve, permite a implementação de clientes em hardware de capacidade restrita
- Sua flexibilidade possibilita o suporte a diversos cenários de aplicativo para dispositivos e serviços de IoT.

MQTT

- Terminologia
- Client - pode ser qualquer dispositivo que se conecte a um broker - de um microcontrolador para um servidor.
- Broker - é responsável por rotear, filtrar e distribuir mensagens para as partes apropriadas (clientes).
- Topic - cada mensagem publicada por um cliente é enviada para um tópico específico no broker.
- QoS - O Quality of Service especifica como o aplicativo deve tentar publicar mensagens em um broker MQTT.

MQTT no Node-RED: Eclipse Mosquitto



MQTT

QoS - Quality of Service

- O MQTT especifica os seguintes três níveis de entrega de mensagens:
- QoS 0: Enviar apenas uma vez.
- Método fire and forget
- Nível de mensagem padrão que não garante a entrega de mensagens.
- Por exemplo, o envio de dados do sensor em QoS 0 pode resultar em perda de medidas. É incomum, mas pode acontecer.

MQTT

- QoS - Quality of Service
- QoS 1
- Entregue pelo menos uma vez.
- Esse nível garante que a mensagem seja entregue, mas não impõe a entrega única.
- QoS 2
- Entregue exatamente uma vez: este nível garante a entrega correta.
- Um caso de uso comum para QoS 1 e QoS 2 é a distribuição da configuração inicial ou das informações do canal para os dispositivos recém-conectados.

Instalação e Execução - Windows

- <https://nodered.org/docs/getting-started/windows>
- Executar o Terminal de Comandos do Windows
- Instalar NPM & NODE.JS
 - Instalar o Node.js
 - Baixar e executar o instalador: <https://nodejs.org/en>
 - Verificar a versão do Node.js instalado: **node --version && npm -v**
- Instalar NODE-RED
 - `npm install -g --unsafe-perm node-red`
- Executar o Node-Red
 - `C:\Users\user> node-red`
- Acessar o GUI do Node-Red pelo Navegador:
 - `https://localhost:1880`

DÚVIDAS?

KOFUJI@USP.BR