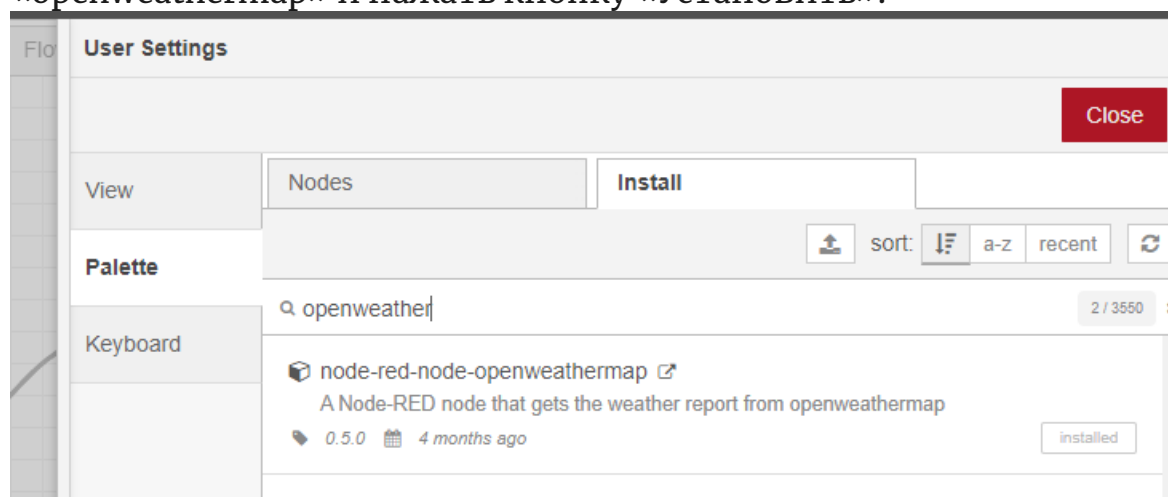


Практическая работа 1(1 часть) Применение Node-Red для построения КФС на основе технологий интернета вещей

Предупреждение о погодных условиях

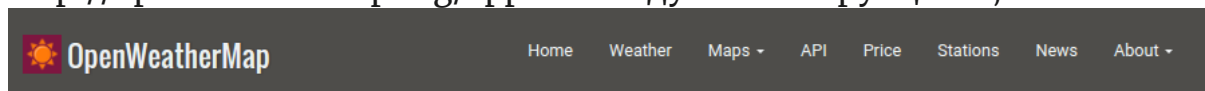
В этом задании вы будете следить за погодой в выбранном населённом пункте и отправлять электронное письмо, когда погодные условия будут меняться. Вы будете использовать узел погоды — openweathermap — который извлекает погоду с сайта openweathermap.org для местоположения, которое вы выберете. Простой пример функционального узла будет использоваться для проверки «ясной погоды» или «плохой погоды», а электронная почта будет использоваться для отправки вам электронного письма при изменении погоды.

1. Ноды «openweathermap» управляются через palette manager Node-RED, доступный в меню Node-RED в правом верхнем углу. Выберите «palette manager», затем «Установить». Оттуда вы можете выполнить поиск «openweathermap» и нажать кнопку «Установить».



Сначала вам нужно будет получить ключ API на OpenWeatherMap. OpenWeatherMap предлагает сервис, предоставляющий подробную информацию о погоде по всему миру. Откройте сайт

<http://openweathermap.org/appid> и следуйте инструкциям, как показано на скриншотах ниже.



How to work with API key

[Home](#) / [API](#) / How to work with API key

To get access to weather API you need an API key whatever account you chose from Free to Enterprise.

How to get API key (APPID)

- 1 Register on the [Sign up](#) page
- 2 Get unique API key on your personal page

How to use API key in API call

Description:

To get access to weather API you need an API key whatever account you chose from Free to Enterprise.

We keep right to not to process API requests without API key.

API call:

`http://api.openweathermap.org/data/2.5/forecast/city?id=524901&APPID={APIKEY}`

Parameters:

APPID {APIKEY} is your unique API key

Example of API call:

`api.openweathermap.org/data/2.5/forecast/city?id=524901&APPID=1111111111`

Вам нужно будет зарегистрироваться в OpenWeatherAccount, как показано ниже:

Create New Account

☐ I agree to the [Terms of Service](#) and [Privacy Policy](#)

Create Account

После регистрации вы будете перенаправлены на свою домашнюю страницу, где сможете получить доступ к своему API-ключу или повторно сгенерировать его. Это должно выглядеть примерно так:



🔍 Weather in your city

Guide

API

Pricing

Maps

Our Initiatives

Partners

Blog

Marketplace

tretiserge ▾

Support ▾

New Products

Services

API keys

Billing plans

Payments

Block logs

My orders

My profile

Ask a question

You can generate as many API keys as needed for your subscription. We accumulate the total load from all of them.

Key

Name

e7eef3785aec401c686a13e2de4d205b

Default



Create key

API key name

Generate

Product Collections

Current and Forecast APIs

Historical Weather Data

Weather Maps

Subscription

How to start

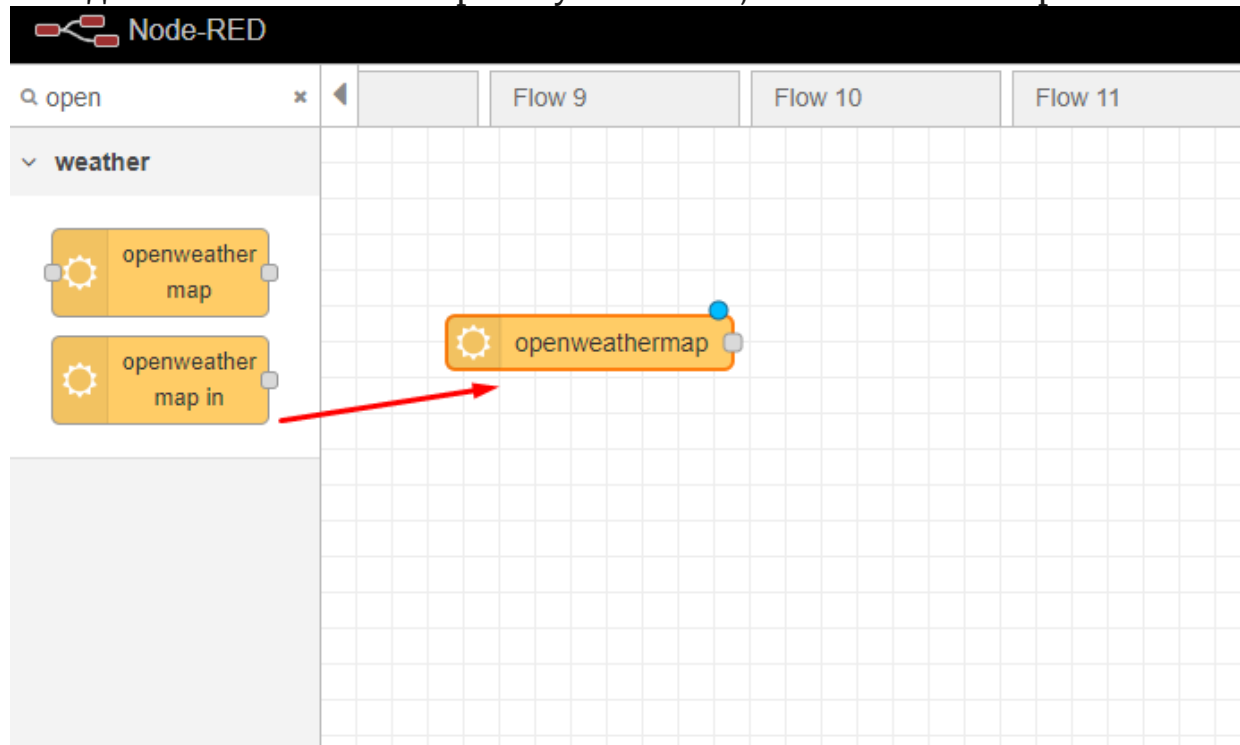
Pricing

Subscribe for free

About us

OpenWeather is a team of IT experts and data scientists that has been practising deep weather data science since 2014. For each point on the

Теперь давайте воспользуемся этим ключом API для создания флоу предупреждений о погоде. Перетащите ноду погоды с левой панели на рабочую область, как показано на рис. 2.11.



Перетащите ноду погоды на пустую рабочую область.

2. Если вы выберете информационную панель справа, вы увидите описание ноды openweathermap с подробной информацией о том, как её настроить и использовать. Некоторые интересные вещи на заметку:

- Нода имеет полную структуру JSON в качестве `msg.payload` с довольно большим количеством сведений о погоде, представленных в виде пар имя: значение, например, скорость ветра и температура.

- Нода определяет 3 новых свойства сообщения: `msg.location`, `msg.time` и `msg.data`. Как упоминалось выше, вы можете свободно добавлять свойства к сообщениям, и узел `openweathermap` добавил эти новые свойства для переноса дополнительной информации, связанной с сообщением.

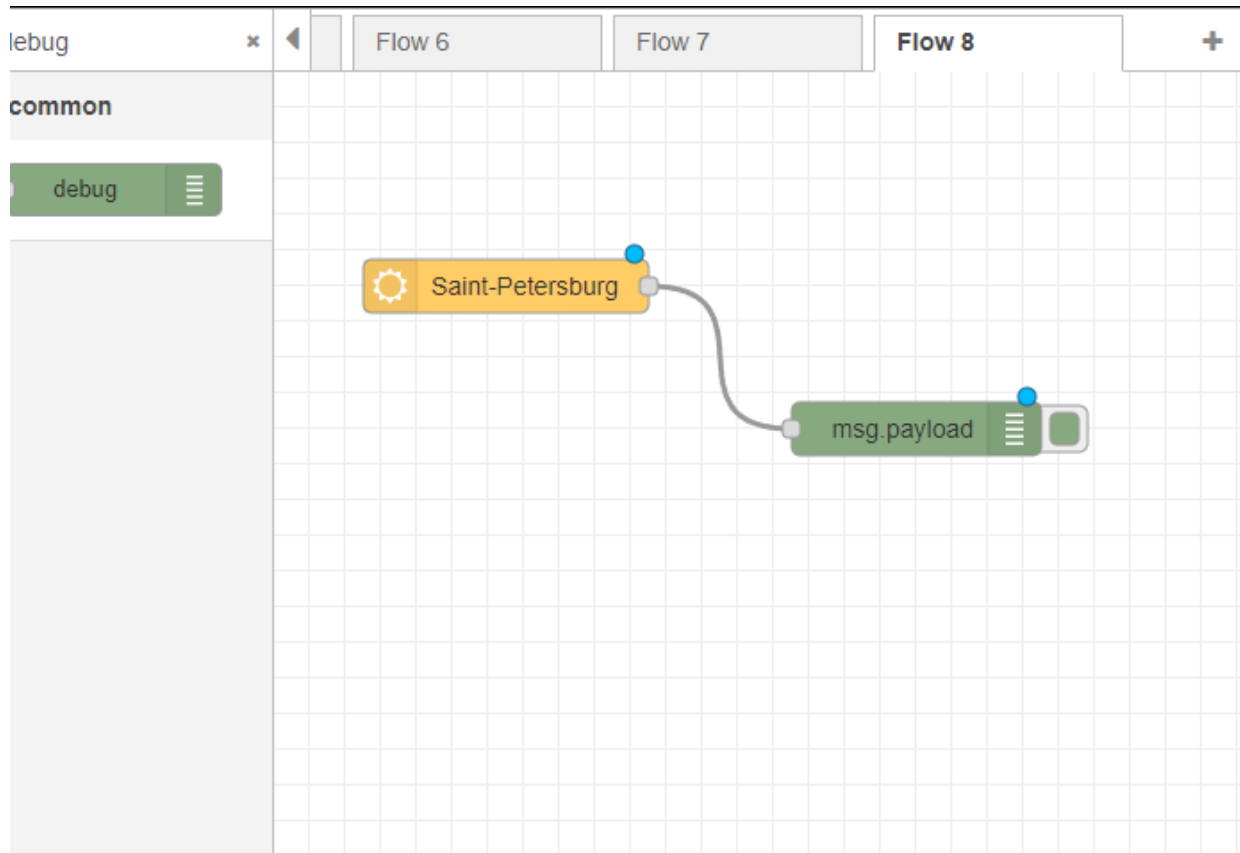
Давайте настроим ноду и посмотрим на фактическую структуру данных, которую он генерирует после запроса вашей локальной погоды. Начните с двойного щелчка по ноде и заполните форму, указав свое местоположение.

Введите свой город и страну в поля. Добавьте ключ API, полученный с <http://openweathermap.org/appid>, и

The image shows a dialog box titled "Edit openweathermap in node". At the top, there are three buttons: "Delete", "Cancel", and "Done". Below these is a "Properties" section. The "API Key" field is highlighted with a red border and a red arrow pointing to it. The "Language" dropdown is set to "English". The "Current weather for" dropdown is set to "Current weather for". The "Location" dropdown is highlighted with a red arrow and is set to "City, Country". Below this, there are two input fields: "City" and "Country". At the bottom, there is a "Name" input field.

нажмите «Готово».

Укажите желаемый город и страну в форме конфигурации, используя ключ API, который вы получили на шаге выше.



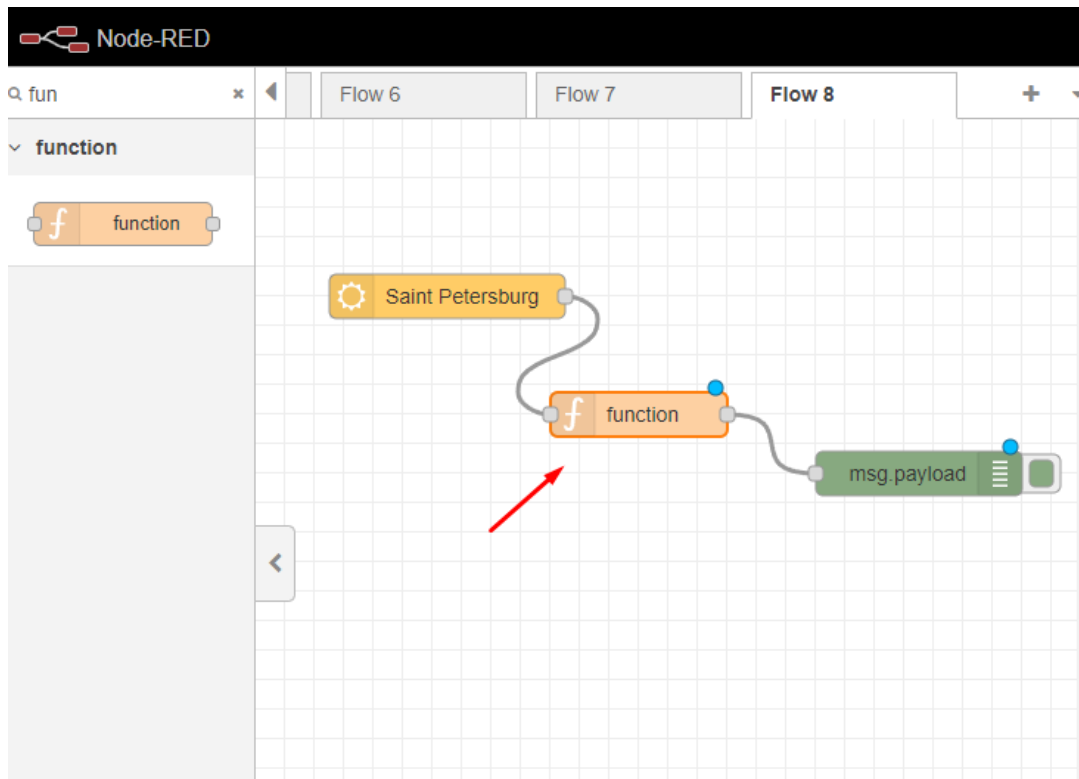
Затем перетащите ноду отладки (debug) и подключите её к ноде openweathermap. Нажмите «Развернуть», чтобы увидеть сообщение msg.payload на панели отладки.

The screenshot shows the Node-RED web interface. At the top, there are tabs for 'Flow 6', 'Flow 7', and 'Flow 8'. The main workspace contains a flow with two nodes: an orange 'Saint Petersburg' node and a green 'msg.payload' node, connected by a wire. On the right side, there is a 'debug' console. The console shows the current flow's message payload as a JSON object. The JSON object contains various weather-related fields for Saint Petersburg, including temperature, humidity, pressure, and a description of the weather conditions.

```
11.11.2021, 22:56:35 node: 4ee0b06d326ef9b1
msg.payload : Object
  object
    id: 804
    weather: "Clouds"
    detail: "overcast clouds"
    icon: "04n"
    tempk: 277.98
    tempc: 4.8
    temp_maxc: 6.1
    temp_minc: 2.7
    humidity: 90
    pressure: 1004
    maxtemp: 279.32
    mintemp: 275.94
    windspeed: 0.89
    winddirection: 158
    location: "Saint Petersburg"
    sunrise: 1636609571
    sunset: 1636637993
    clouds: 90
    description: "The weather in Saint Petersburg at coordinates: 59.8944, 30.2642 is Clouds (overcast clouds)."
```

Message payload для ноды openweathermap представляет собой структуру JSON, описывающую погодные условия, температуру, ветер, облачность и время восхода солнца.

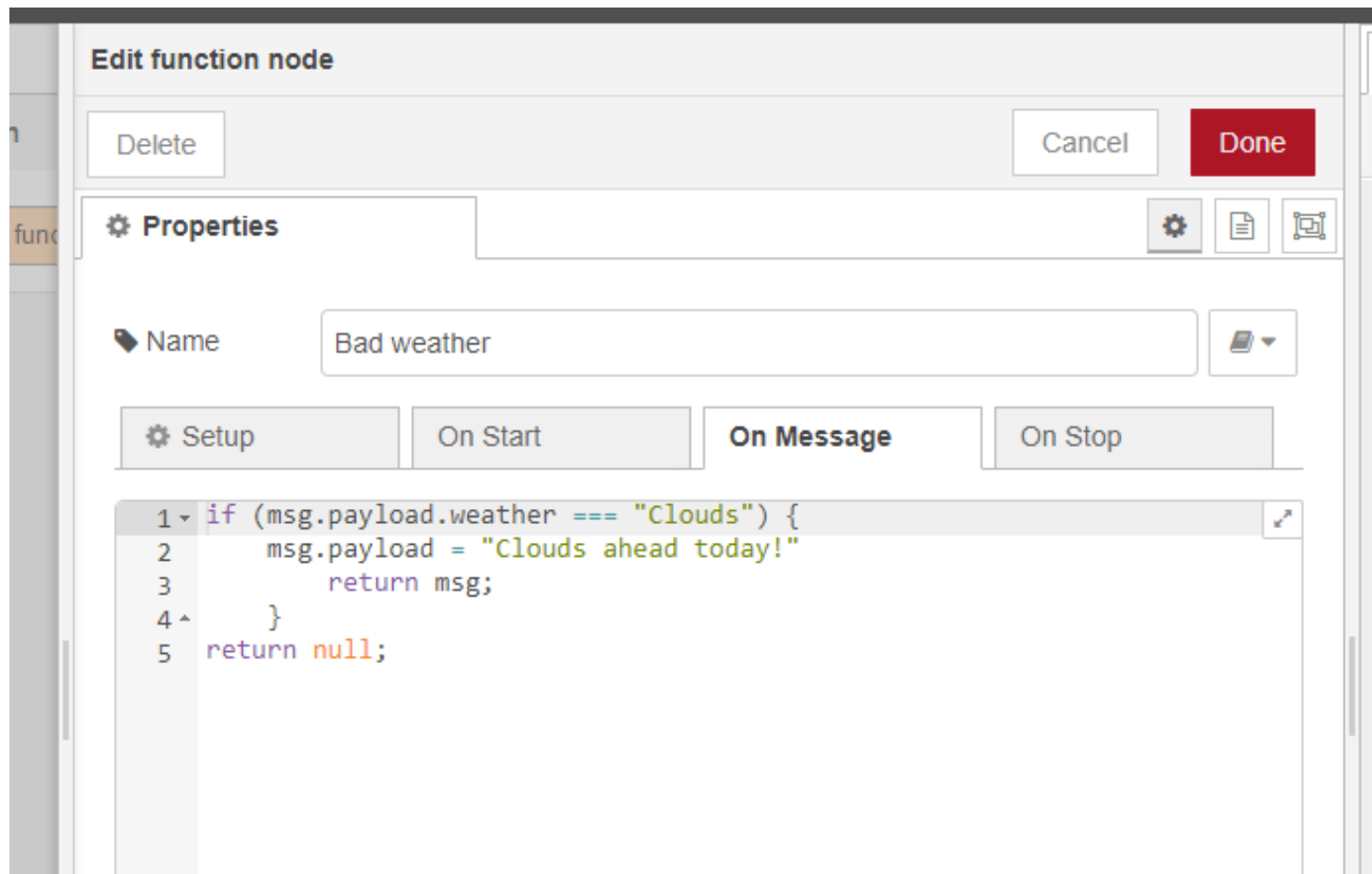
3. Как вы можете видеть, нода предоставляет довольно много информации о погоде в виде обычных пар имя:значение. На данный момент мы должны проверить значение «weather» и если там указаны, например, «clouds», то необходимо отправить предупреждение об облачной погоде. Для этого мы будем использовать function node.



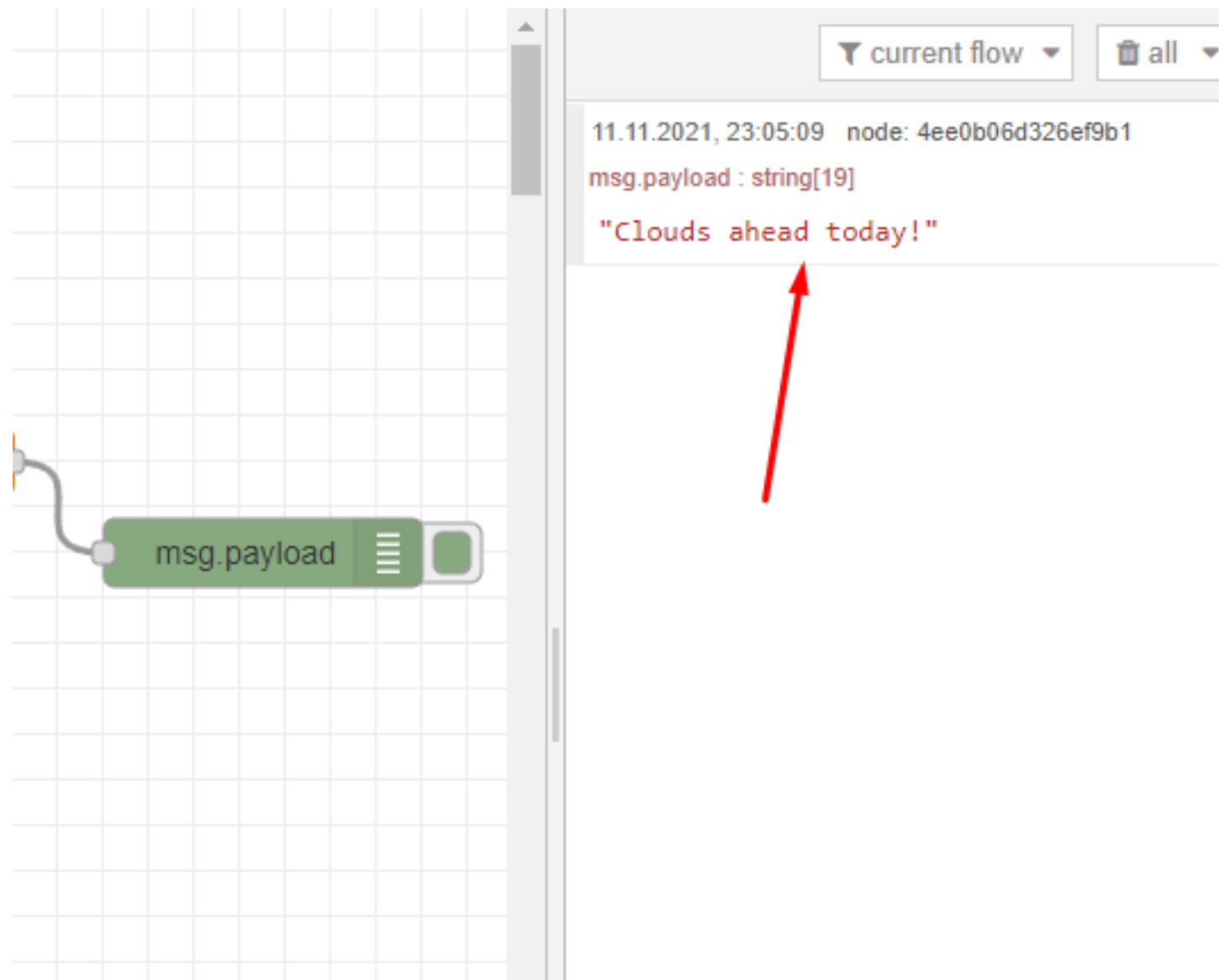
Необходимо сделать двойной клик на function node и вставить следующий код:

Function node “If Bad Weather”

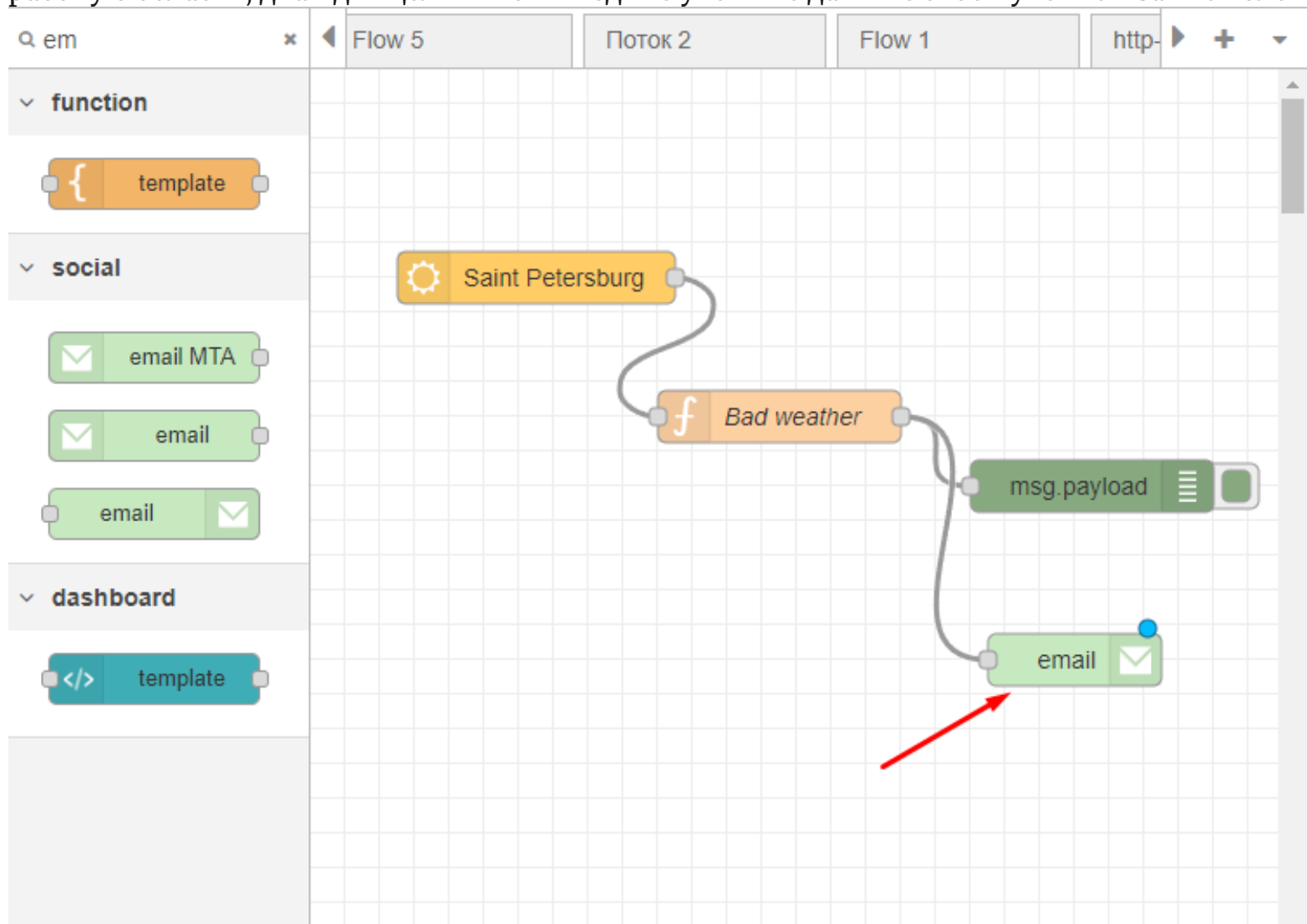
1. `if (msg.payload.weather === “Clouds”) {`
2. `msg.payload = “Clouds ahead today!”`
3. `return msg;`
4. `}`
5. `return null;`



Проанализировав код, вы можете увидеть, что он анализирует payload message входящего сообщения для параметра погоды и сравнивает его со строкой «Clouds» (строка 1). Если он равен, он переписывает payload сообщения вашей собственной строкой «Clouds ahead today!» (строка 2). В противном случае будет возвращено нулевое сообщение (строка 5). Этот последний бит важен, потому что узлы Node-RED игнорируют нулевые сообщения.



4. Теперь мы будем использовать ноду вывода электронной почты. Перетащите ноду электронной почты в рабочую область, дважды щелкните и введите учетные данные своей учетной записи электронной почты.



Edit email node

Delete Cancel Done

Properties

To tretiserge@mail.ru

Server smtp.mail.ru

Port 465 ☒ Use secure connection.

Userid tretiserge@mail.ru

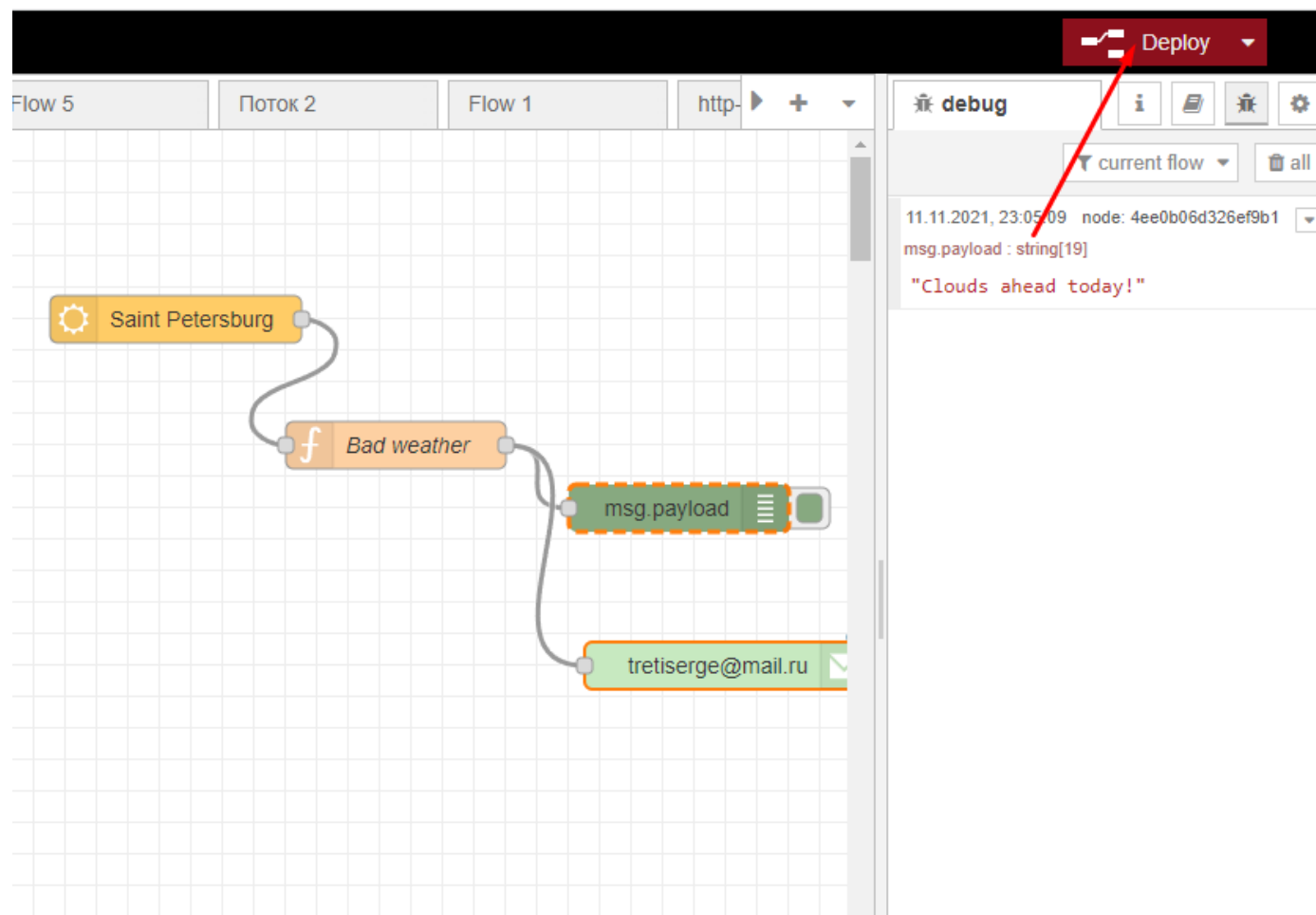
Password

Use TLS? ☒

Name Name

После того, как вы соединили весь flow, вы можете нажать кнопку Deploy, а затем посмотреть свою учетную запись электронной почты, чтобы видеть новое электронное письмо каждый раз, когда облачно.

Нажмите “Deploy”:






Сгруппируйте письма себе в отдельную папку, чтобы они всегда были под рукой!



[Включить](#)

Current Weather Information

-  tretiserge@mail.ru Сегодня, 23:10
Кому: вам



Clouds ahead today!



Ответить



Переслать



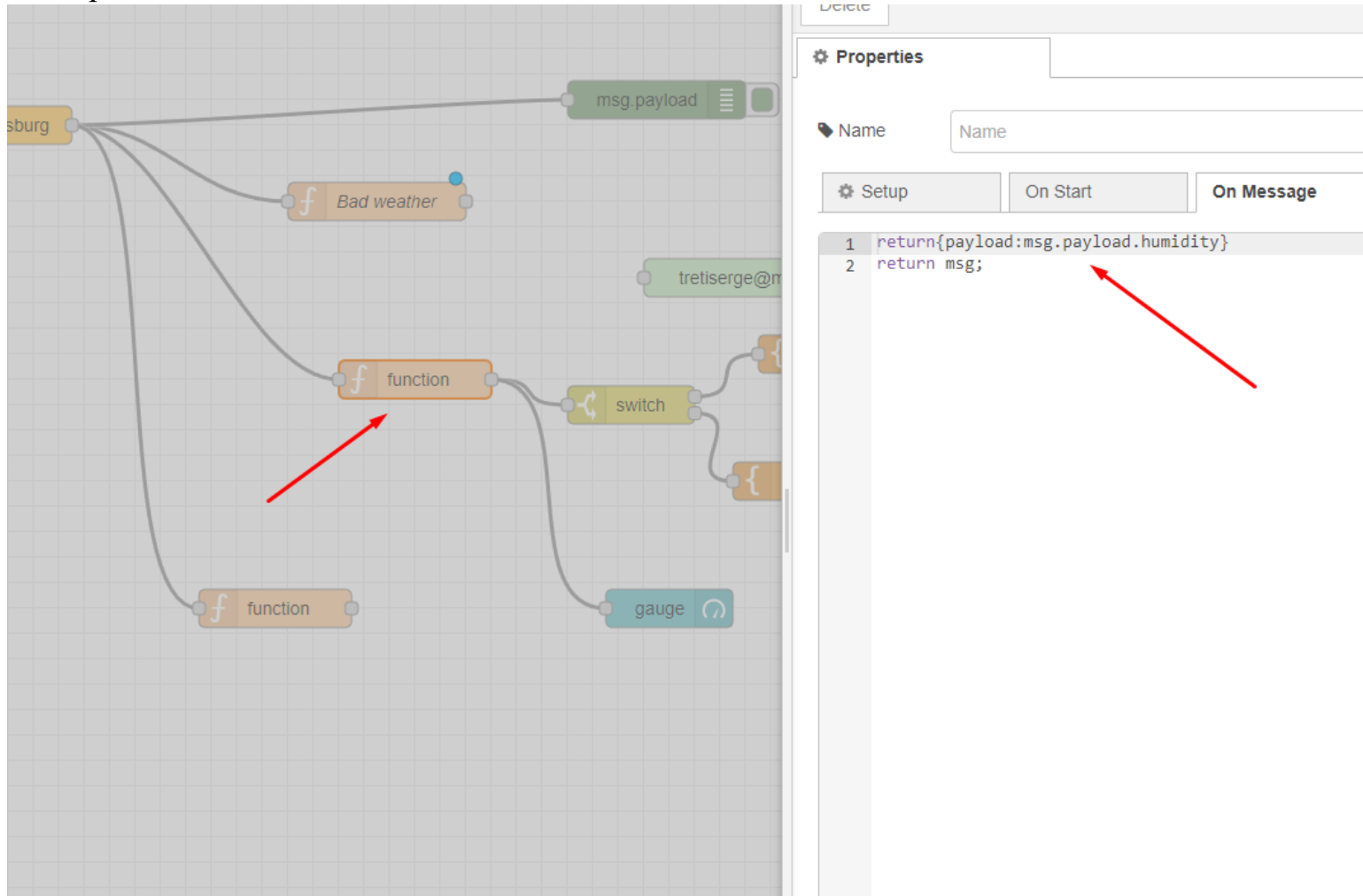
Предложить звонок



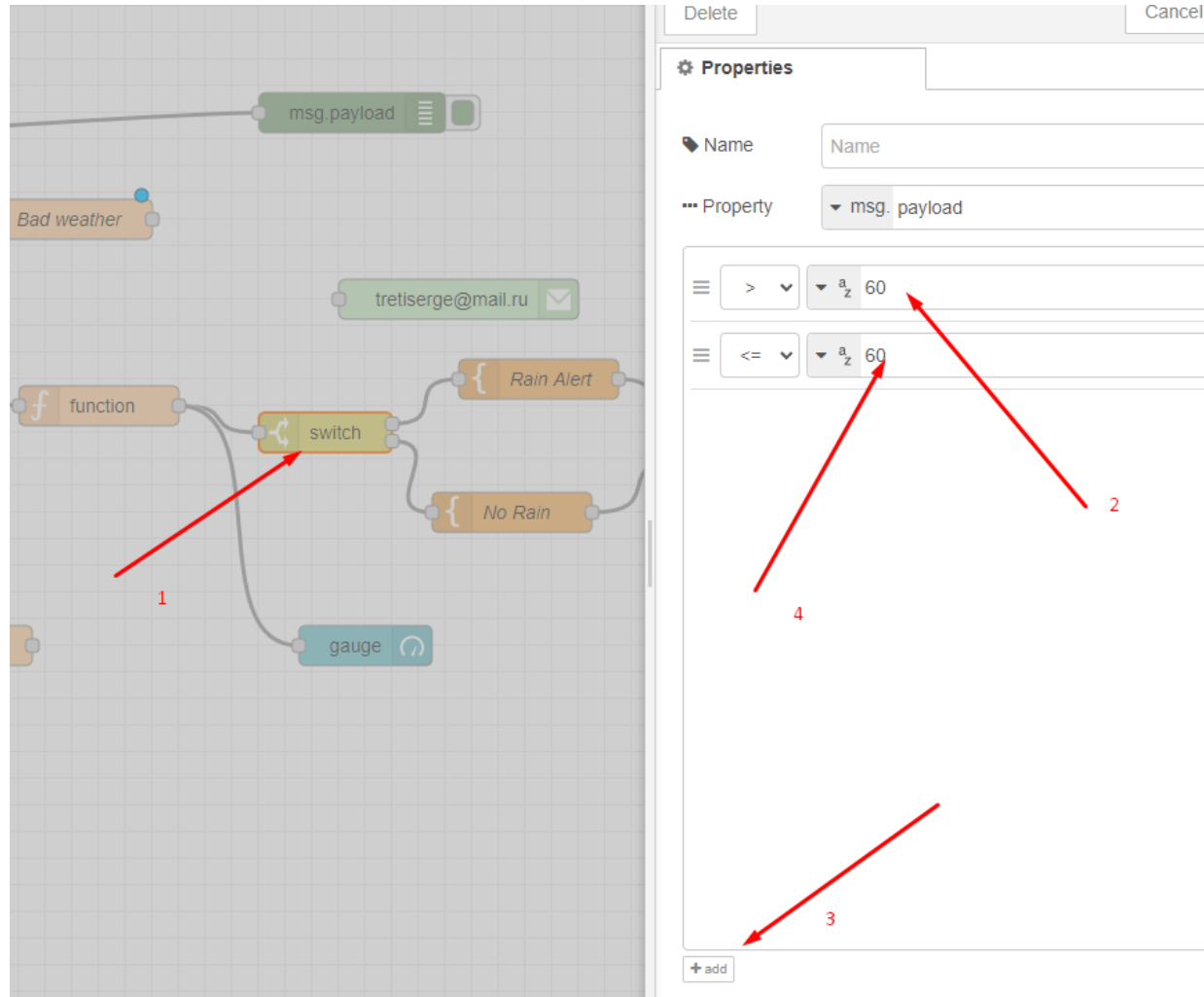
Защищено
kaspersky

5. Теперь сделаем предупреждение о дожде. Это означает, что если влажность более 60%, то велика вероятность дождя, а если меньше – дождя сегодня не будет. Мы будем использовать ноды switch и template.

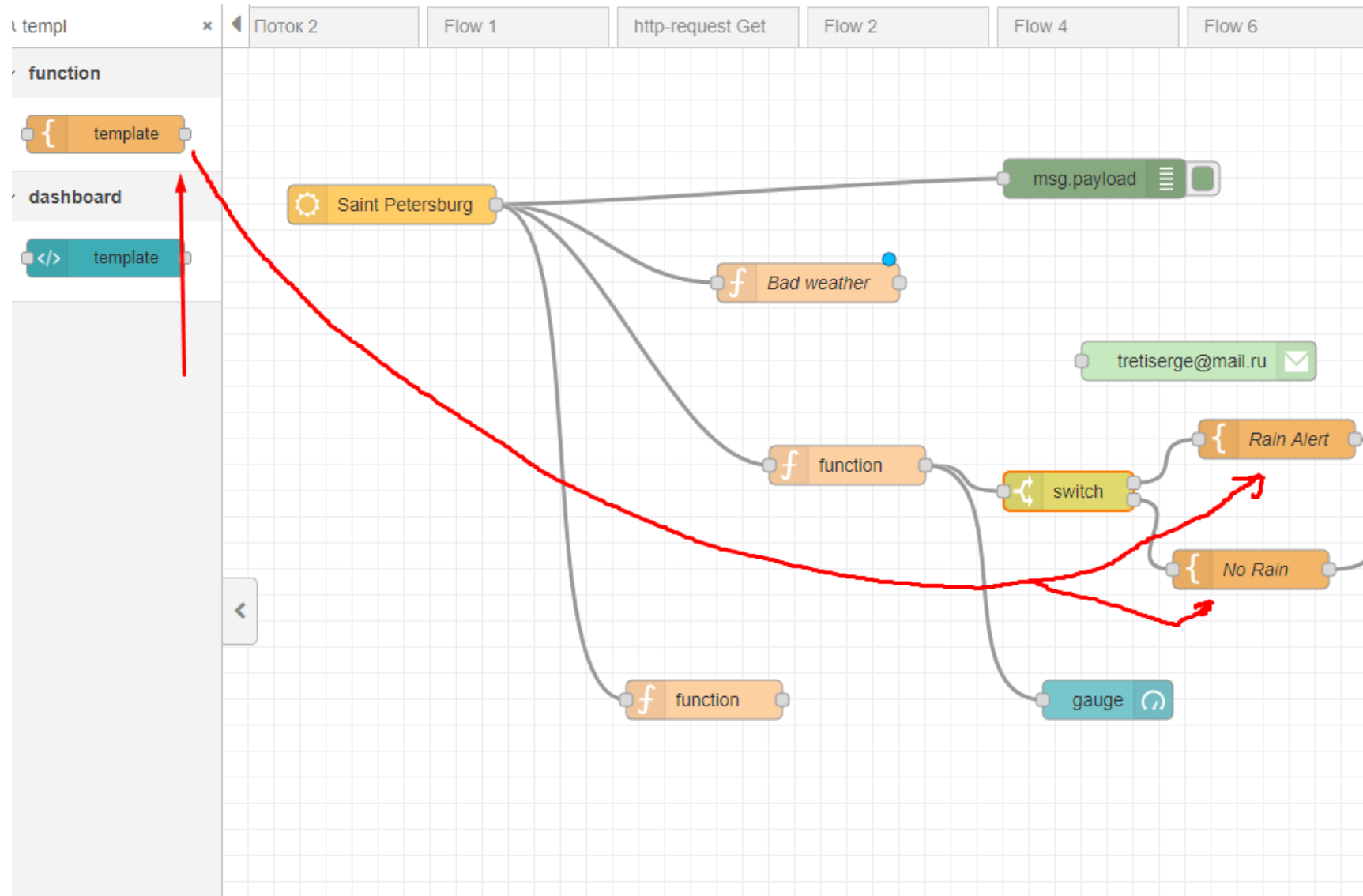
Во-первых, мы добавим новый function нод для влажности.

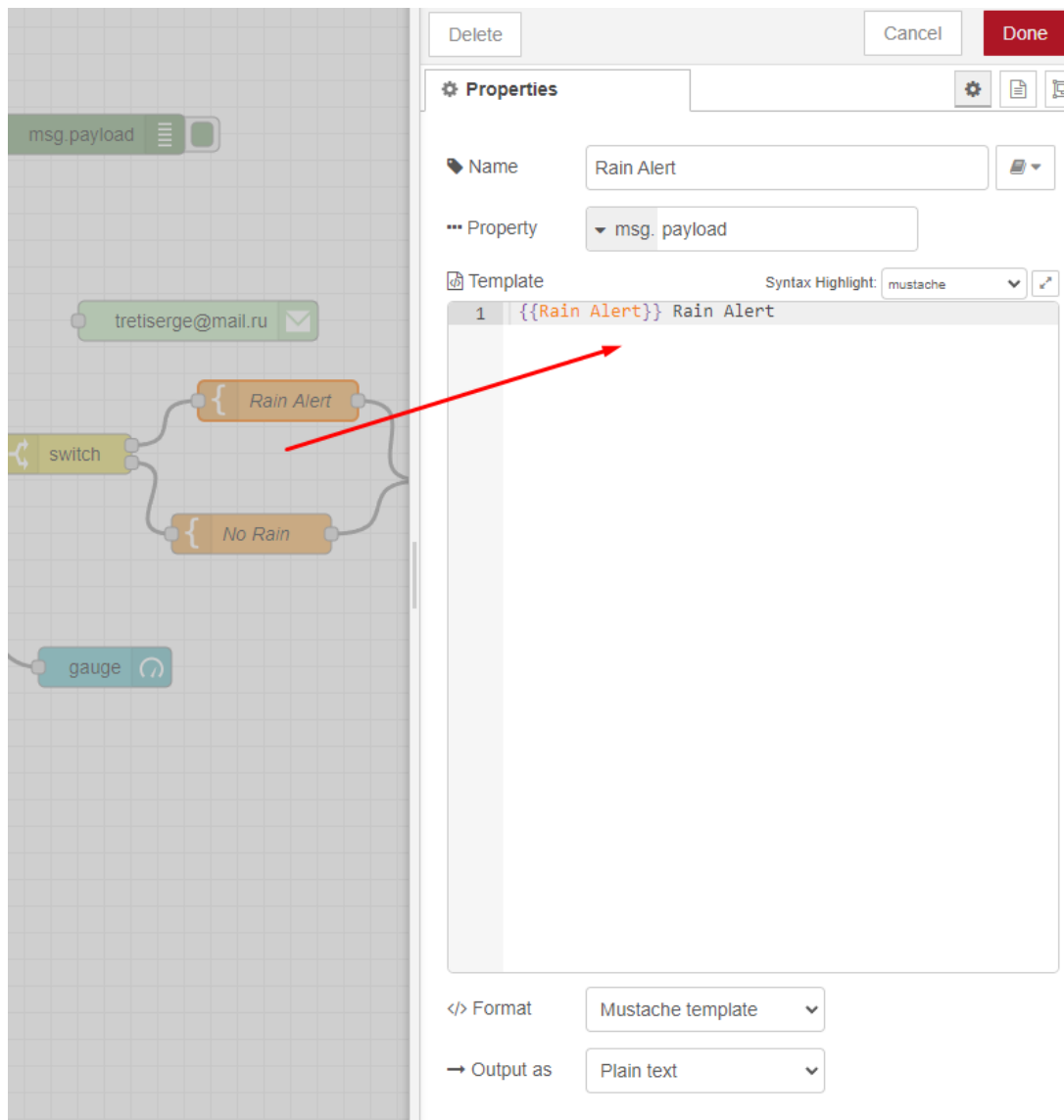


Затем необходимо добавить switch node для разделения значений влажности.

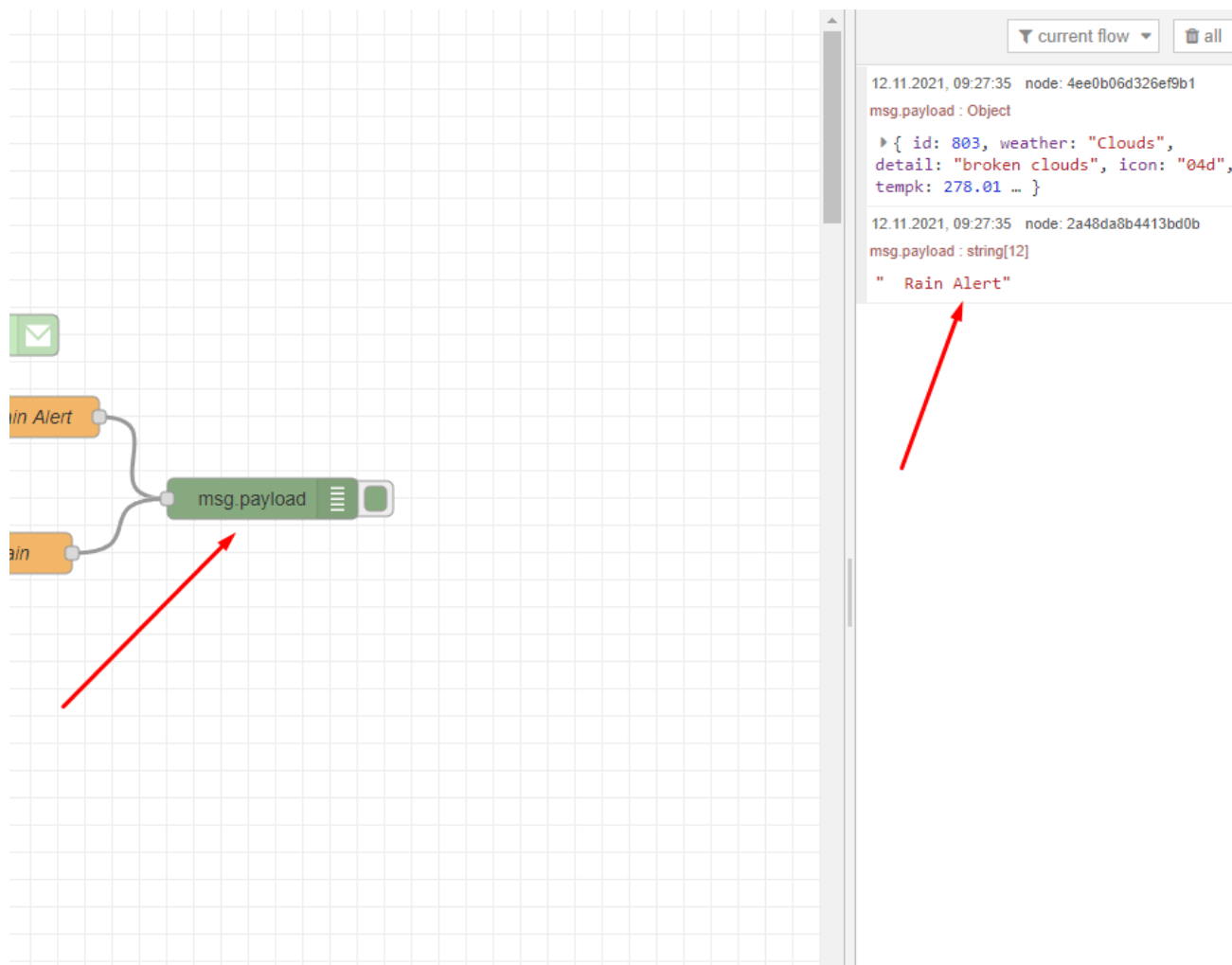


Теперь добавим две ноды template для разделения сообщений:





И последний шаг – добавить debug node, нажать deploy и посмотреть результат в окне отладки.



Как альтернатива – можно сделать тоже самое используя только function node.

Edit function node

Delete

Cancel

Done

⚙ Properties

📌 Name

humidity

⚙ Setup

On Start

On Message

On Stop

```
1 if (msg.payload.humidity >= "70"){  
2     msg.payload = "Rain alert!"  
3     return msg;  
4 }  
5 msg.payload = "No rain today ";  
6 return msg;
```

▶

+

▼

debug

i

▼

▼ current flow ▼

all ▼

21.11.2021, 18:16:16 node: 3ae8249e73398030

msg.payload : string[11]

"Rain alert!"

Практическая работа 1(2 часть) Изучение протокола MQTT

1. Получение сообщения в виде JSON объекта через MQTT сервис

Эта часть практической работы основана на ноде MQTT, которая предоставляет собой удобный инструмент Node-Red для получения данных от брокера MQTT. Для тех, кто не знаком с MQTT, это пример системы публикации/подписки (обычно сокращенно до системы публикации/подписки), которая позволяет датчикам публиковать обновления, которые доставляются клиентам, подписанным на этот датчик. MQTT использует модель темы, позволяющую издателям (например, датчикам) создавать темы и публиковать данные в темах. Точно так же другие могут подписаться на тему и получать асинхронное уведомление о данных, опубликованных в теме.

Системы Pub/Sub — это отличный способ подключения слабосвязанных распределенных систем, и они хорошо соответствуют типичным шаблонам киберфизических и IoT систем, когда устройства или вещи генерируют события, которыми вы хотите поделиться. Протокол MQTT, помимо того, что он асинхронный, также является легким и не требует таких высоких пропускных каналов, как, например, HTTP, что для устройств с ограниченными ресурсами часто является важным преимуществом. MQTT был первоначально разработан в конце 1990-х годов и использовался в различных условиях IoT. MQTT стал стандартом в 2014 году и является стандартной частью многих наборов инструментов IoT. MQTT на самом деле расшифровывается как Message Queuing Telemetry Transport.

Чтобы использовать узел mqtt, вам необходимо иметь доступ к брокеру. Существует ряд работающих бесплатных серверов MQTT, например <http://test.mosquitto.org/>, или тот, который будет использоваться в этой лаборатории, www.hivemq.com. Используя адрес брокера и тему, вы можете настроить входной узел mqtt для подписки на эту тему, заставляя его генерировать новое сообщение всякий раз, когда в этой теме публикуются новые данные. Сообщение будет содержать информацию об опубликованных данных, включая сами данные в msg.payload и топик MQTT-брокера в msg.topic.

Чтобы начать работу с узлом mqtt, вы будете использовать бесплатный брокер mqtt hivemq, который доступен по адресу (<http://www.hivemq.com/demos/websocket-client/>). Конечно, вы можете использовать любого MQTT-брокера, в том числе своего собственного, если он у вас установлен.

Сначала перетащите входную ноду mqtt и настройте его для брокера. Не забудьте настроить тему на что-то уникальное, в этом примере мы используем noderedlab/sensor, но вы должны использовать свою собственную уникальную тему для отчета, то есть <ваше имя>/sensor

mqtt

Flow 4

Flow 1

network

mqtt in

mqtt out

noderedlab/sensor

connected

Edit mqtt in node

DeleteCancelDone

Properties

Serverbroker.mqttdashboard.com:1883

ActionSubscribe to single topic

Topicnoderedlab/sensor

QoS2

Outputauto-detect (string or buffer)

NameName



4 Edit mqtt in node > Edit mqtt-broker node

Delete

Cancel

Update

⚙ Properties

⚙ 📄

🔖 Name

Name

Connection

Security

Messages

🌐 Server

broker.mqtdashboard.com

Port

1883

☒ Connect automatically

☐ Use TLS

⚙ Protocol

MQTT V3.1.1

▼

🔖 Client ID

Leave blank for auto generated

💓 Keep Alive

60

ℹ Session

☒ Use clean session



Connection

● connected



Host

broker.mqttdashboard.com

Port

8000

ClientID

clientId-GiiuX7IBml

Disconnect

Username

Password

Keep Alive

60

SSL



Clean Session



Last-Will Topic

Last-Will QoS

0

Last-Will Retain



Last-Will Message

Publish



Topic

noderedlab/sensor

QoS

0

Retain



Publish

Message

{"analyze":false,"value":10}

Subscriptions



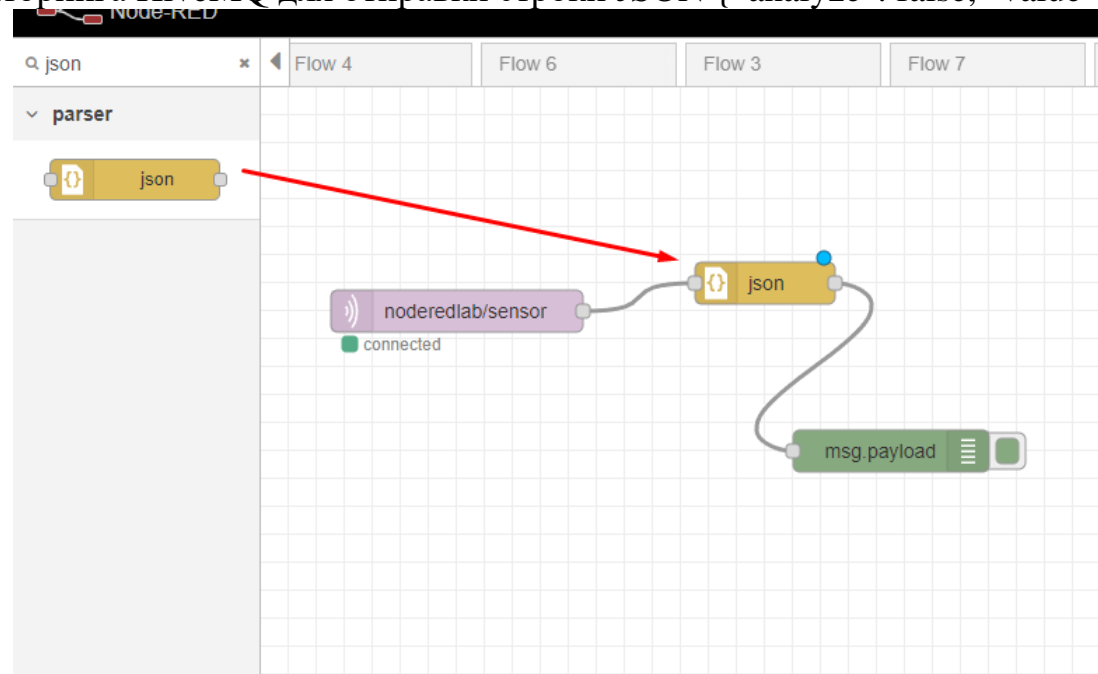
Add New Topic Subscription

В этой практической работе вы будете использовать клиент HiveMq WebSocket, поэтому перейдите на эту страницу и подключитесь к брокеру. Вы опубликуете закодированную строку JSON в теме, которую вы настроили, чтобы увидеть использование ноды mqtt и ноды json.

Поскольку вы отправляете строку JSON, вам нужно будет проанализировать сообщение, которое генерирует нода mqtt, когда она получает сообщение MQTT. Для этого вам нужно перетащить ноду json и подключить её к выходу ноды mqtt.

Нода json Node-RED представляет собой своего рода функцию для удобства, поскольку она анализирует входящее сообщение и пытается преобразовать его в/из JSON. Поэтому, если вы отправите ему строку JSON, она преобразует ее в объект JavaScript и наоборот.

Если вы подключили обычный debug node к ноде json и выполнили Deploy, то необходимо использовать панель мониторинга HiveMQ для отправки строки JSON `{"analyze": false, "value": 10}`.



Last-Will Topic

Last-Will QoS

0

Last-Will Retain

☐

Last-Will Message

Publish

Topic

noderedlab/sensor

QoS

0

Retain

☐

Message

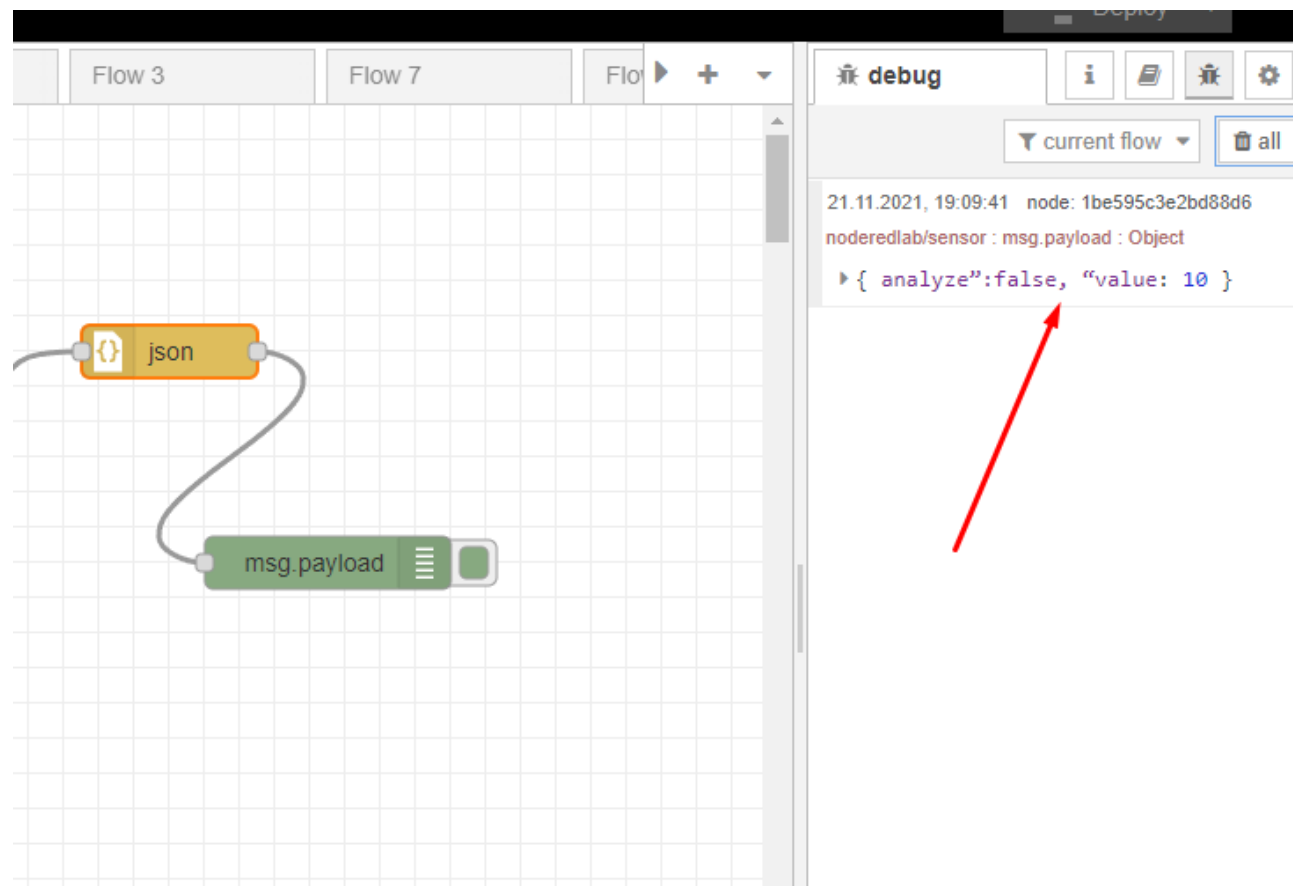
{"analyze":false,"value":10}

Publish

Subscriptions

Add New Topic Subscription

Messages



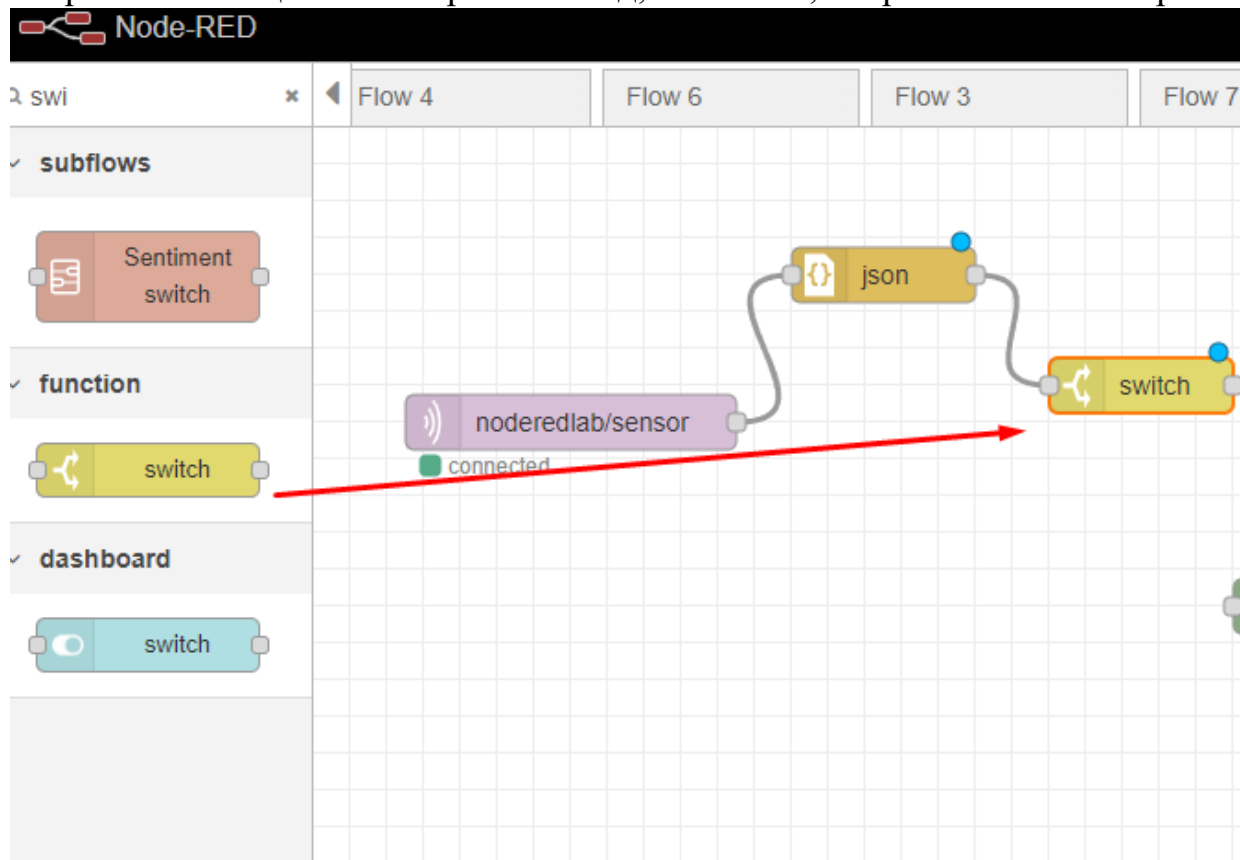
Итак, вы получаете и анализируете сообщение MQTT, отправленное в виде строки JSON.

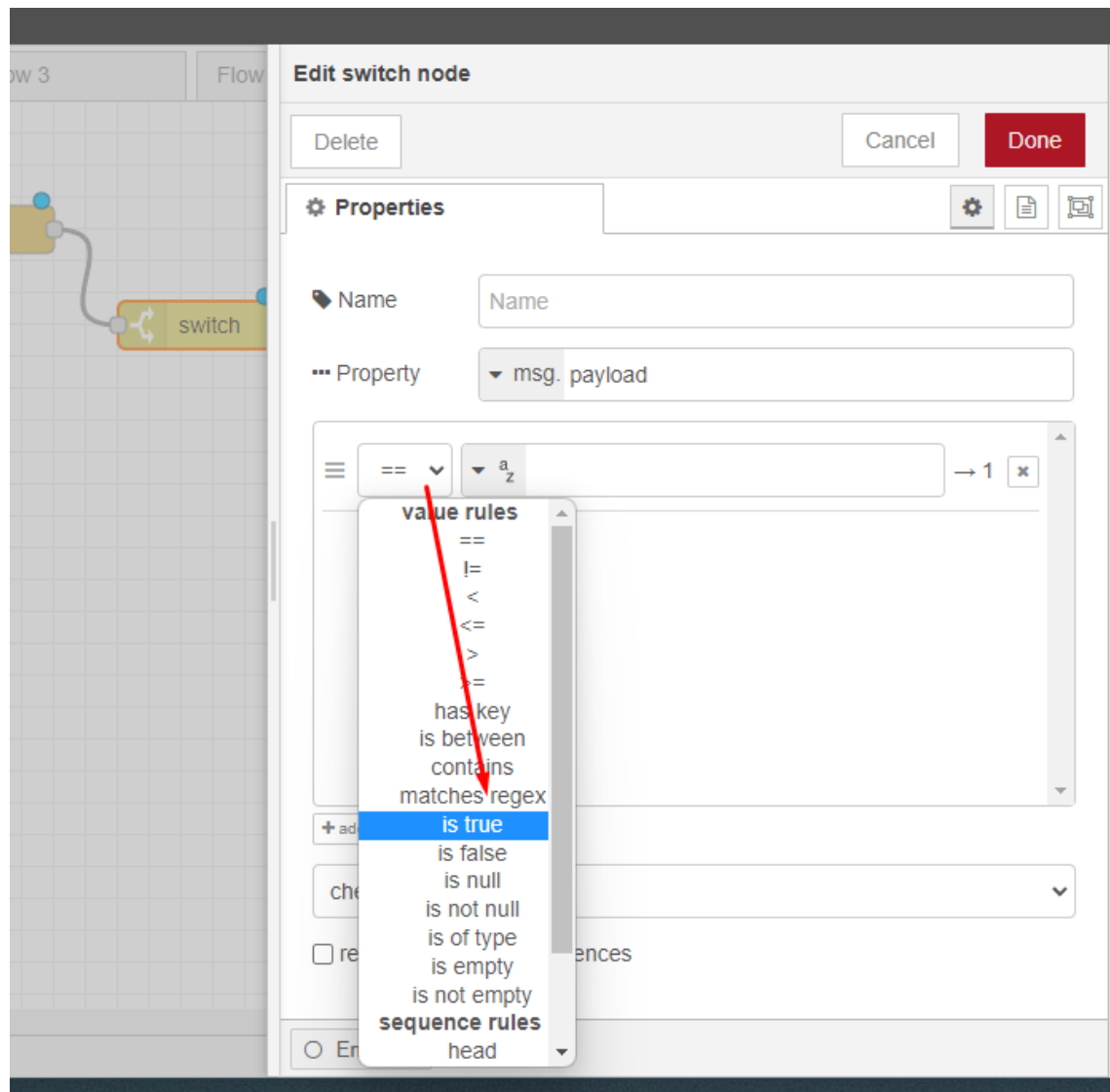
Если вы внимательно посмотрите на вывод, то увидите, что `msg.payload` содержит объект, который сам имеет два поля: `analyze` и `value`, каждое со своими собственными значениями. Вы можете получить доступ к этим полям через `msg.payload.analyze` и `msg.payload.value`. Давайте посмотрим на ноду, которая поможет это сделать.

2. Использование switch node для анализа JSON объекта

Одной особенностью объекта JSON является то, что вы можете легко изменять его свойства. Полезной нодой Node-Red для этого является switch node. Её роль заключается в «переключении» или маршрутизации сообщений в зависимости от свойств входящего сообщения. Например, вы можете проверить свойство `msg.payload.analyze` и, в зависимости от его значения (`true/false`), принять решение о маршрутизации сообщения на один из выходов switch node.

Перетащите switch node и дважды щелкните по ней. Настройте её для оценки свойства «`msg.payload.analyze`». Если `true`, отправить сообщение на первый выход; если `false`, отправить его на второй выход.





Edit switch node

Delete

Cancel

Done

⚙ Properties

⚙

📄

🖨

📁 Name

Name

⋮ Property

▼ msg.

payload.analyze

☰

is true

▼

→ 1

✕

☰

is false

▼

→ 2

✕

✚ add

checking all rules

▼

☐ recreate message sequences

☐ Enabled

Теперь вы можете подключить две ноды debug — когда вы настраиваете несколько выходов для ноды, они нумеруются сверху, поэтому выход 1 — это верхний выход, а выход 2 — нижний.

Last-Will Message

Publish

Topic: QoS: Retain: ☐

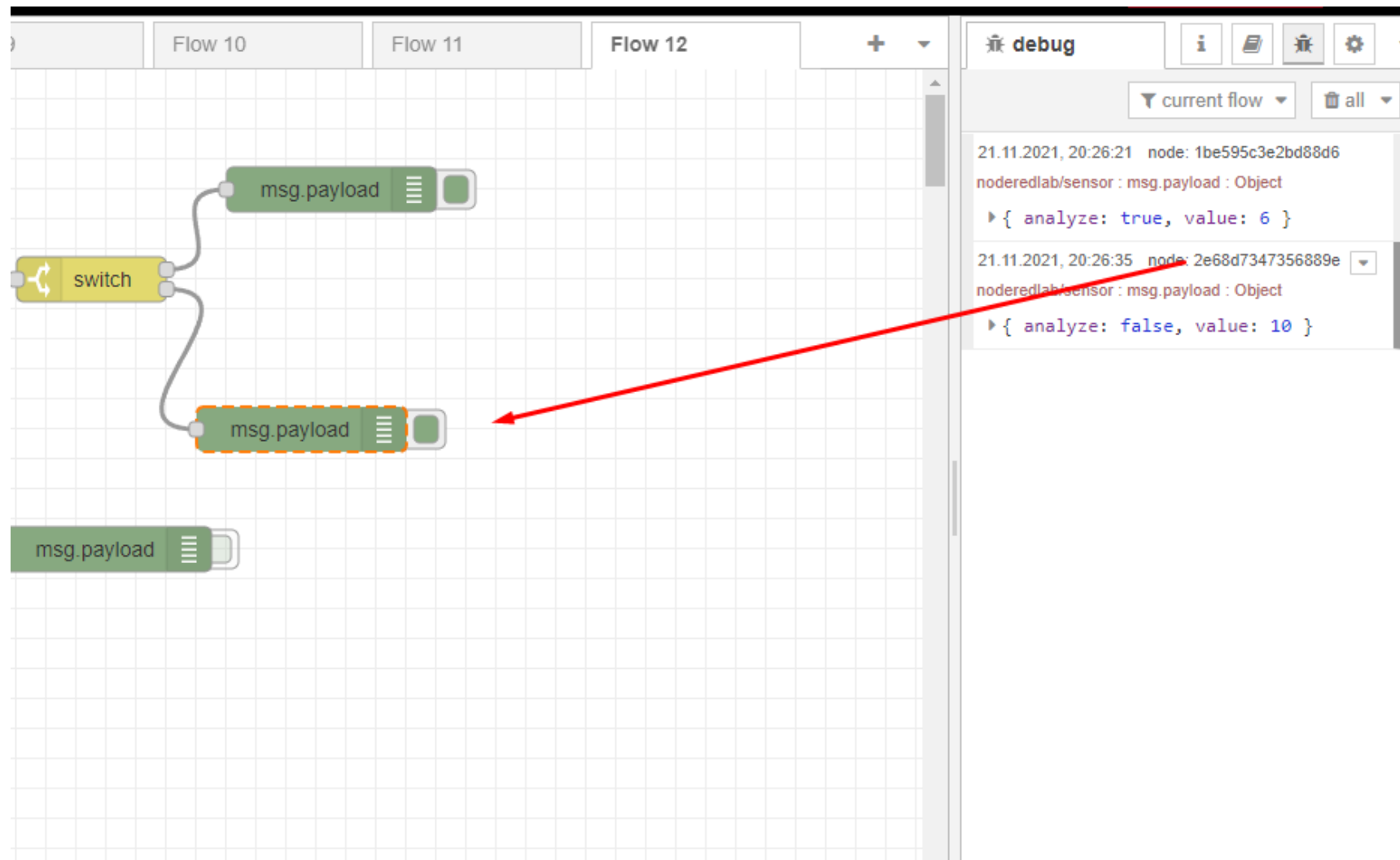
Message: **Publish**

Messages

Subscribe

Если теперь вы вернетесь на страницу ввода HiveMQ и отправите сообщение MQTT {"analyze": true, "value": 6}, вы увидите, что первый (верхний) вывод активирован и входящие сообщения маршрутизируются, или 'Switch», на выход 1. Если вы отправите исходное сообщение {«analyze”:false, “value”:10}, узел переключения активирует выход 2, а исходный debug node работает. Наведение указателя на сообщение отладки покажет, какой debug node выводит сообщение.

Как видите, это дает вам встроенный нод Node-RED, который позволяет быстро определять содержимое входящих сообщений и направлять сообщение в разные части потока в зависимости от ввода.



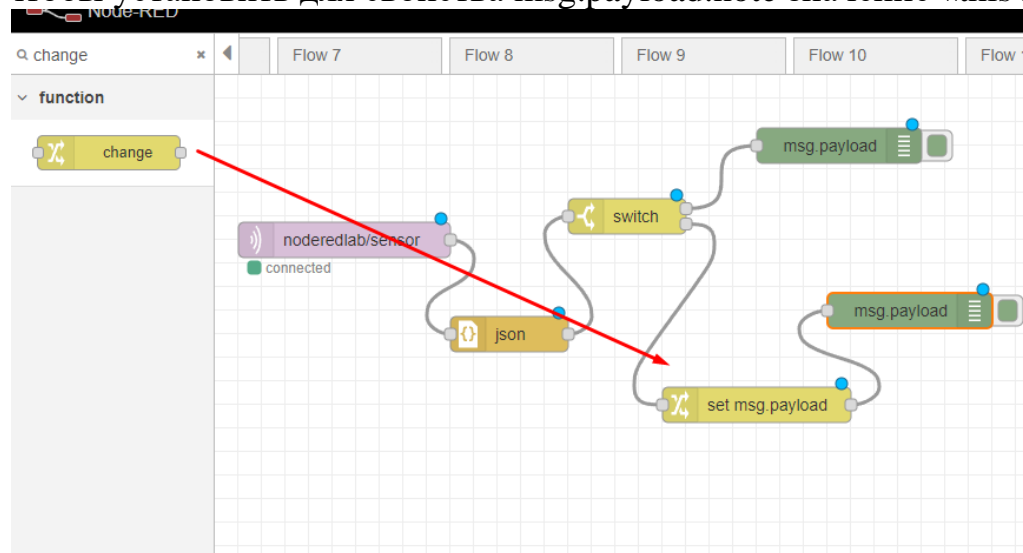
3. Использование Change node для изменения или управлением message payload сообщения

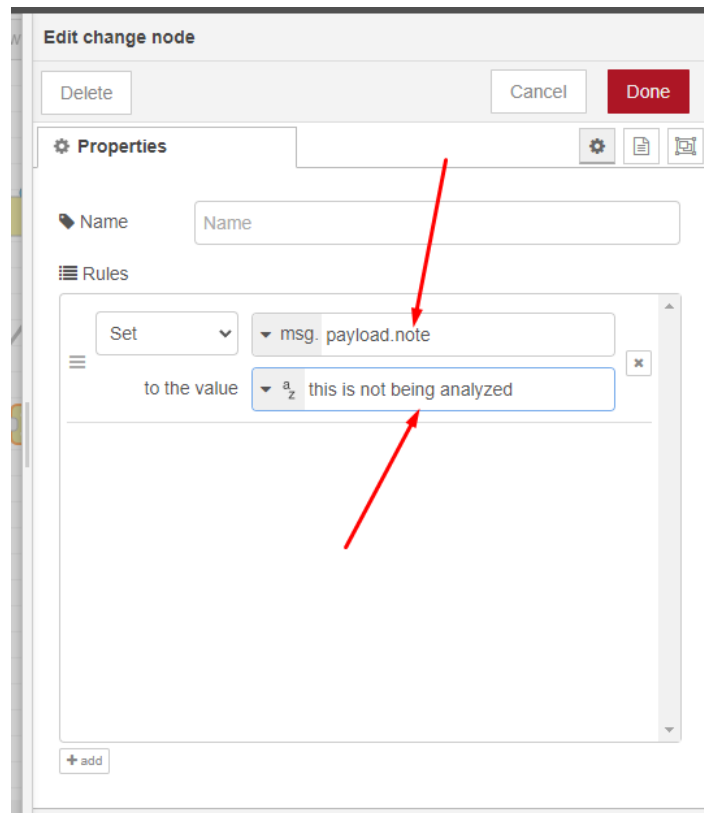
Еще одной полезной нодой является Change node, которая позволяет вам изменить message payload сообщения или добавить новые свойства. Вы можете использовать эту ноду, чтобы влиять на свойства в сообщении, изменяя существующие, удаляя их или добавляя новые свойства.

В этой части практической работы 1 вы продолжите тему MQTT и увидите, как теперь, когда вы успешно «переключили» поток сообщений на основе входящего сообщения MQTT, вы можете добавить новое свойство сообщения msg.payload.note.

Во-первых, давайте перетащим change node и подключим его ко второму выходу switch node. Как вы помните, это вывод, который срабатывает, когда для msg.payload.analyze установлено значение false.

Теперь настройте его, чтобы установить для свойства msg.payload.note значение «this is not being analyzed».





Когда вы получите сообщение, которое switch node отправляет на 2-й выход, оно будет изменено, чтобы содержать элемент «note» со строкой «this is not being analyzed». Если вы запустите и протестируете флоу, отправив сообщение MQTT из HiveMQ, вы увидите вывод:

Deploy

Flow

debug

i

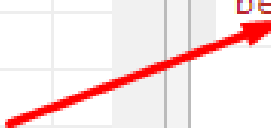
current flow

all

21.11.2021, 20:35:23 node: 2e68d7347356889e

noderedlab/sensor : msg.payload : Object

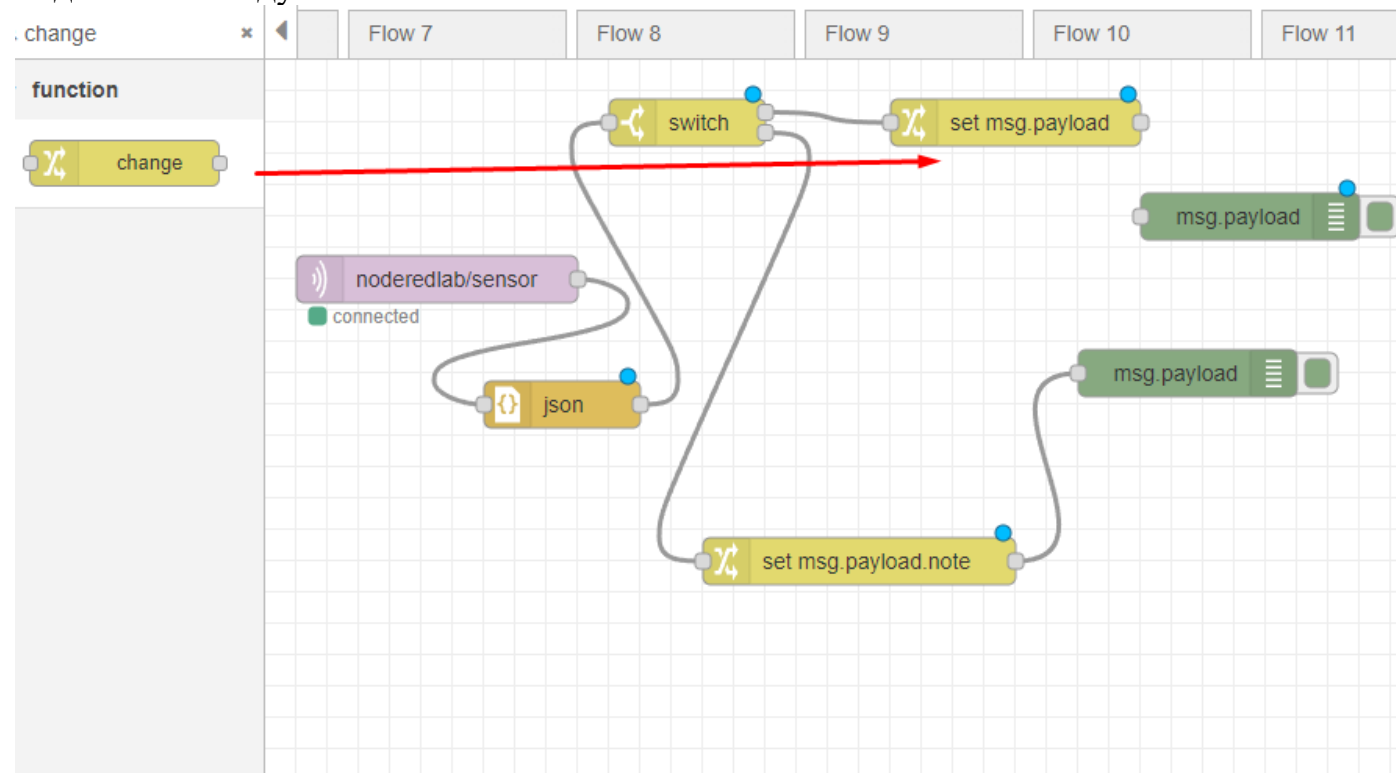
▶ { analyze: false, value: 10, note: "this is not being analyzed" }



4. Использование ноды rbe (отчет по исключению)

В этой части работы вы добавите ноды в часть флоу, которые используются, когда вы решаете, что флоу должен быть дополнительно проанализирован. Вы будете использовать ноду rbe (отчет по исключениям), которая передает данные только в том случае, если они изменились. Вы можете настроить её так, чтобы она проверяла message payload сообщения и либо блокировался до тех пор, пока сообщение не изменится (режим rbe), либо пока сообщение не изменится на указанную величину (deadband mode). В режиме rbe работают с числами и строками. В режиме deadband mode она работает только с числами и использует настроенную зону пропускания как положительных, так и отрицательных значений, так что входящее значение может колебаться в пределах заданного диапазона, прежде чем оно сработает.

Сначала давайте добавим еще один change node, который вы подключите к выходу 1 switch node. Затем вы подключите ноду rbe к switch node.

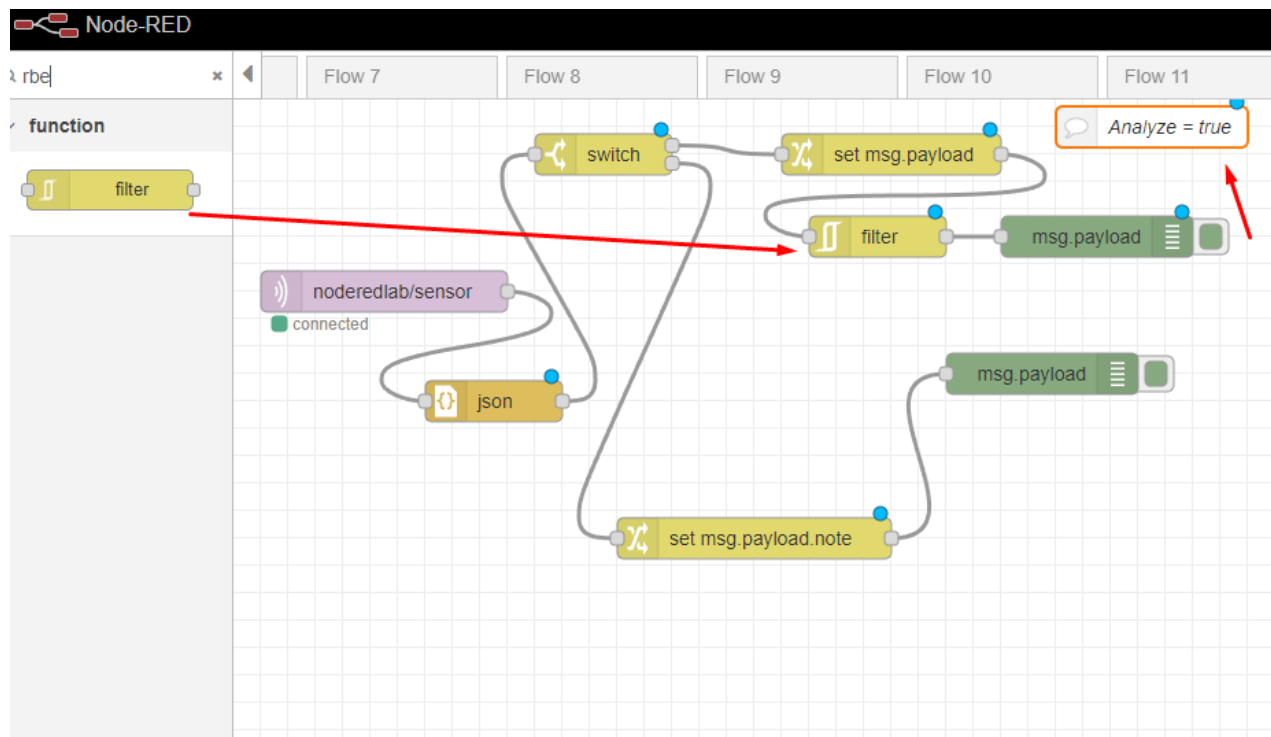


Давайте подключим узел `change` и узел `rbe` как показано ниже. Чтобы напомнить нам, что этот вывод имеет дело с флагом «analyze», добавьте `comment node` и напишите «Analyze = true». Комментарии полезны при написании сложных флоу.

Добавьте теперь ноду `rbe` для проверки того, изменились ли наши входные данные более чем на 20%

Отредактируйте `change node`, чтобы установить для `msg.payload` значение `msg.payload.value`. Это установит вывод этой ноды в значение, найденное в элементе `msg.payload.value` полученного на входе.

Поскольку вы хотите определить, изменилось ли это значение на 20 % или более, вам нужно дважды щелкнуть узел `rbe` и настроить его на блокировку, если значение не изменится более чем на 20 %.



Edit change node

Delete

Cancel

Done

⚙ Properties

⚙

📄

🖼

🏷 Name

Name

☰ Rules

Set ▾

▼ msg. payload

☰

to the value

▼ msg. msg.payload.value

☐ Deep copy value

The diagram illustrates the configuration of a 'Set' rule in the 'Edit change node' dialog. Three red arrows point to specific fields:

- The first arrow points to the `msg. payload` field in the first rule row.
- The second arrow points to the `msg. msg.payload.value` field in the second rule row.
- The third arrow points to the ☐ Deep copy value checkbox in the second rule row.

9 **Edit filter node**

Delete Cancel Done

Properties

Mode block unless value change is greater than

20% compared to last valid output value

Property msg. payload

☒ Apply mode separately for each

msg. topic

Name Name

Publish

Topic noderedlab/sensor QoS 0 Retain ☐ Publish

Message

```
{"analyze":true,"value":6}
```

debug

current flow

21.11.2021, 21:12:28 node: 1be595c3e2bd88d6
noderedlab/sensor : msg.payload : number
6

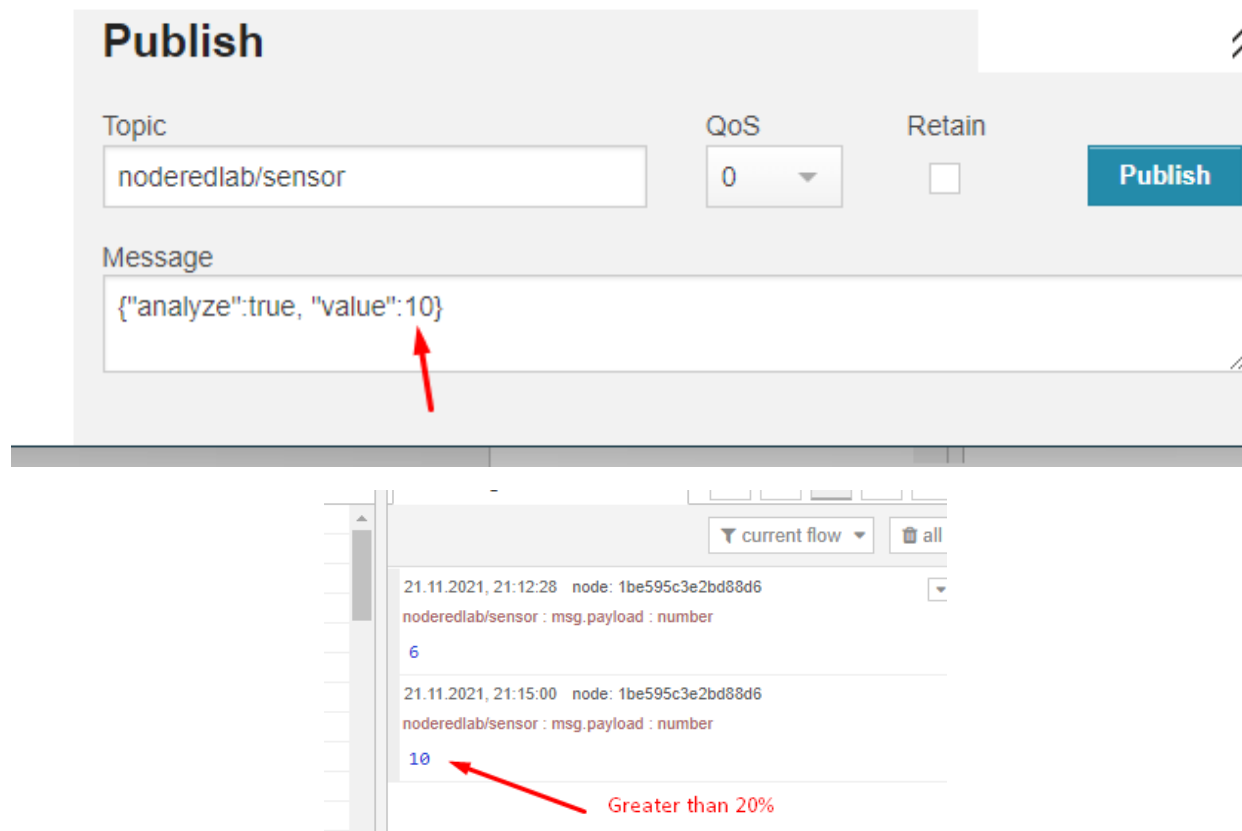
Topic: noderedlab/sensor QoS: 0 Retain: ☐ Publish

Message: {"analyze":true, "value":7}

current flow all

21.11.2021, 21:12:28 node: 1be595c3e2bd88d6
noderedlab/sensor : msg.payload : number
6

No value



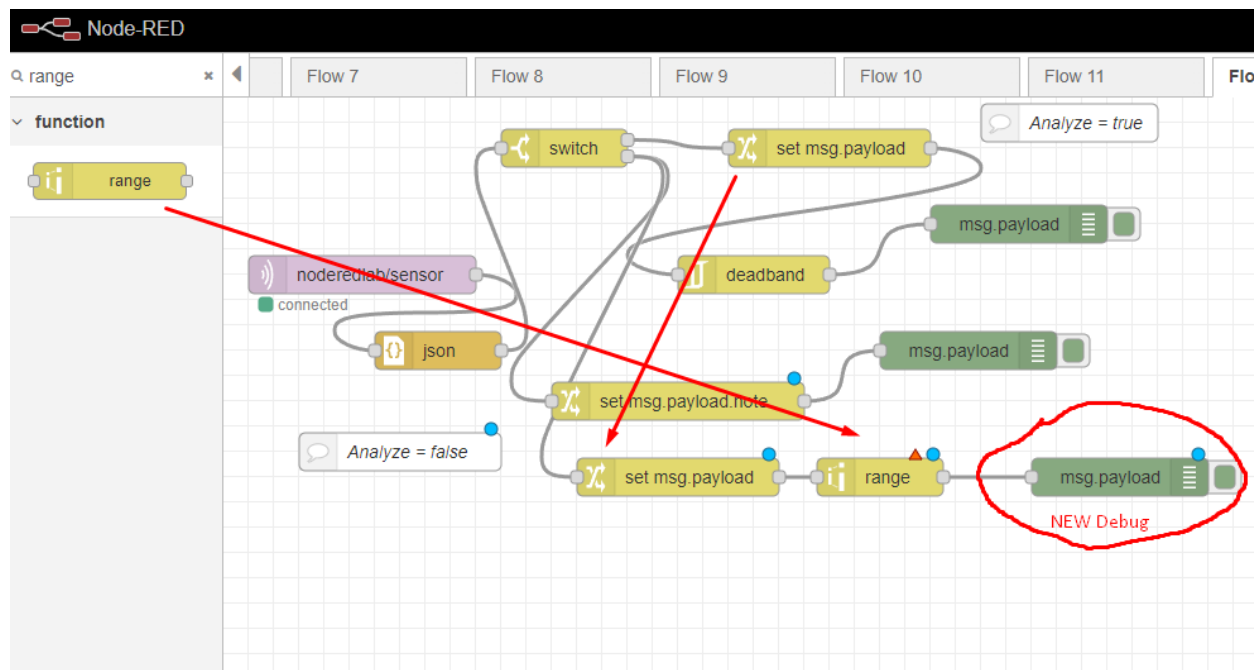
Чтобы протестировать флой, запустите его, а затем вернитесь на страницу HiveMQ и отправьте серию сообщений. Во-первых, вам нужно установить значение analyzer равным true, чтобы switch node отправлял сообщение на выход 1. Если вы используете исходное значение сообщения, равное 7, то rbe node не позволит передать это сообщение. Если вы затем отправите второе сообщение со значением 10, rbe node оценит разницу между 6 и 10, увидит, что она больше 20%, и отправит сообщение, которое будет показано в окне отладки.

5. Масштабирование входных значений при помощи range node

При работе с реальными входными данными от датчиков и других устройств часто требуется возможность масштабирования входных данных. Range node как раз позволяет масштабировать (линейно) входное значение.

Предположим, вы хотите масштабировать свои данные (первоначально в диапазоне 0-10) до диапазона (0-255) только когда никакого анализа не проводится. Это означает, что мы имеем дело с нижней частью нашего флоу, запускаемого, когда switch node оценивает свойство Analyze как false.

Для этого выберите change node, который вы настроили выше (установите msg.payload), и скопируйте его, нажав ctrl+c, затем ctrl+v. Перетащите range node. Дважды кликните на ней и сконфигурируйте как показано ниже (to map the input from 0-10 to 0-255).



Edit range node

Delete Cancel Done

Properties

Property: msg. payload

Action: Scale the message property

Map the input range:

from: 0 to: 10

to the target range:

from: 0 to: 255


☐ Round result to the nearest integer?

Затем вернитесь на тестовую страницу HiveMQ и опубликуйте {"analyze":false, "value":10} в качестве нового сообщения MQTT в той же теме.

Publish

Topic: QoS: Retain: ☐

Message:




Flow

debug

current flow all

21.11.2021, 21:26:13 node: 2e68d7347356889e
noderedlab/sensor : msg.payload : Object
▶ { analyze: false, value: 10, note: "this is not being analyzed" }

21.11.2021, 21:26:13 node: b0781357a69c7bba
noderedlab/sensor : msg.payload : number
255



Если вы вернетесь в окно Node-RED, вы увидите, что debug node, связанный с нижней частью флоу, запущен, показывая, что свойство `msg.payload.value`, которое вы установили равным 10 при публикации в MQTT увеличено до 255.