Министерство образования Хабаровского края
Краевое государственное бюджетное образовательное учреждение
среднего профессионального образования
«Хабаровский колледж межотраслевых технологий и сферы обслуживания»

Изучение среды потокового программирования Node-red и её применения в «Интернете вещей»

Методическое пособие

для специальности 09.02.01 «Компьютерные системы и комплексы»

Разработчик
Мазур Т.В., руководитель
предметно-цикловой
комиссии
информационных
дисциплин,
преподаватель в.к.к.

Хабаровск, 2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В 2022 году был принят новый федеральный образовательный стандарт по специальности 09.02.01 Компьютерные системы и сети. Поменялись многие профессиональные компетенции, а в соответствие с ними и содержание дисциплин и модулей. В современном мире при бурном развитии информационных технологий и вычислительной техники, при внедрении нейросетей и методов искусственного интеллекта в различные сферы жизни, программа обучения студентов по «компьютерным» специальностям не может не меняться. И впервые там стало встречаться упоминание современных контроллеров, «умного» дома, мобильных роботов, интернета вещей и др.

Нам преподавателям тоже нужно осваивать новые технологии и в этом очень помогает федеральная программа 5000 мастеров, в рамках которой было организовано обучение и стажировка по актуальным направлениям, в том числе и связанном с вычислительной техникой.

Мною была выбрана компетенция «Интернет вещей», современное и актуальное направление. IoT упрощает и автоматизирует сложные задачи, которые иногда выходят за рамки человеческих возможностей. Количество подключенных устройств, составляющих IoT, сегодня исчисляется миллиардами.

При анализе информации об этом направлении выходило, что основная платформа, которую преподаватели раньше использовали для реализации идей Интернета вещей, в том числе и на чемпионатах «Профессионалы» - ThingWorx – сейчас недоступна в России. Выяснилось, что преподаватели, эксперты, организаторы и участники чемпионата вынуждены искать новые программные решения. Наилучшим вариантом на данный момент оказалась среда Node-red. Но даже на стажировке, которая проводилась в Федеральном технопарке г. Калуги, было видно, что платформа до конца не изучена и не понята экспертами по компетенции Интернет-вещей. Отсутствовали какие-либо методические материалы, учебники, в том числе электронные.

Материалы всемирной сети предлагались на английском языке в видеоформате. Таким образом, были очевидны методические проблемы в сопровождении и подготовки слушателей.

В тоже время было понятно, что возможности платформы потокового программирования впечатляют, и их можно использовать не только для управления роботами на чемпионате Профессионалы, но и для работы с контроллерами, базами данных, создания веб-интерфейса пользователя для управления объектами автоматизации и во многих других сферах. Node-red может объединить датчики и базы данных для хранения получаемы значений, контроллеры и веб-интерфейс, который используется для управления параметрами, и много другое.

Разработанное мною методическое пособие отражает мои знания и опыт в самостоятельном изучении Node-red, а также в нём присутствует практическая работа, составленная по материалам пройденных курсов, которые проводила эксперт компетенции Интернет вещей Байшагурова Розалия Рифовна (г. Москва) и приведено моё решение задач, которые вошли в демонстрационный экзамен в конце стажировки.

Методическое пособие включает в себя теоретический и практических блок. В теоретическом разделе рассказывается, что такое Интернет вещей, история его возникновения, основные принципы устройства, отличительные черты промышленного интернета вещей. В практической части даны указания по установки и использованию платформы Node-red и её применению в автоматизации, базах данных, Интернете-вещей, взаимодействию с контроллером Atduino, программирование которого изучается в колледже в ПМ02 по специальности 09.02.01, но который мы раньше не связывали с веб-интерфейсом и базами данных.

Интернет вещей

С помощью Интернета можно управлять не только информацией, но и вещами, которые нас окружают в повседневной жизни. Уже давно никого не удивляет объединение в одну сеть нескольких устройств (например, компьютера, планшета и смартфона) или управление через Интернет промышленным оборудованием. Следующий шаг – контроль с помощью возможностей Всемирной сети над бытовыми приборами для большего комфорта обычной жизни. Именно этим занимается Интернет вещей. Приведу несколько определений.

Интернет вещей (IoT, Internet of Things) – это объединение разных устройств в общую сеть, в которой они могут собирать информацию, обрабатывать ее и обмениваться данными между собой, с человеком и серверами в дата-центре или облаке.

IoT (интернет вещей, Internet of Things) – это взаимодействие различных устройств между собой и окружающим миром, исключающее участие человека и позволяющее изменить некоторые социальные и экономические нормы.

IoT – это технология вычислительной сети физических предметов («вещей»), оснащённых встроенными технологиями для взаимодействия друг с другом или с внешней средой.

Промышленный Интернет Вещей (Industrial Internet of Things, IIoT) – адаптированная к потребностям промышленности версия IoT, основе которой тщательный сбор и глубокий анализ данных о физических параметрах объектов, взаимодействующих в процессе производства, возможность получения данных от любого подключенного устройства или интеллектуального датчика из любого корпоративного приложения, независимость от конкретного оператора связи, возможность одноранговой связи между подключенными устройствами, использование

стандартизованных протоколов на базе IP-сетей, применение программного обеспечения (ПО) промежуточного слоя и облачных технологий.

История возникновения Интернета вещей

Предвестником идеи создания Интернета вещей был Никола Тесла, который высказывался на эту тему ещё в 1926 г. По его мнению, радио, совершенствуясь, постепенно превратится в «огромный мозг», а все прочие инструменты и устройства будут подключаться к нему и станут столь компактными, что поместятся в карман.

Автором первой «Интернет-вещи» был Джон Ромки, входивший в число разработчиков и изобретателей TCP/IP. В 1990 г. ему удалось подключить обычный кухонный тостер к ПК и заставить этот прибор включаться и выключаться по команде, отправленной с компьютера (Get и Set). Это позволило управлять тостером дистанционно и даже программировать его на автономную работу.

А вот термин был придуман только в 1999 г. Однако особо не использовался, поскольку до 2010-х гг. в области Интернета вещей не было никаких особых прорывов, она не развивалась. Что в общем неудивительно: в то время только настольные ПК, смартфоны, ноутбуки и серверы имели все нужные интерфейсы, чтобы законнектиться с Интернетом, и достаточную вычислительную мощность.

Да и не было особой потребности в компьютеризации бытовой техники и подключении её к Интернету — это только сделало бы её дороже.

По мере того, как распространялись беспроводные технологии, микросхемы дешевели и мир повсеместно глобализировался, о концепции интернета вещей стали вспоминать всё чаще. Микропроцессоры ARM, чья энергоэффективность превзошла процессоры для десктопов, становились всё популярнее.

Ещё одна важная веха в истории интернета вещей относится к 2009 г. Количество подключенных к интернету «вещей» стало больше, чем всё население Земли. И таких устройств будет ещё больше — электронная начинка сегодня появляется практически везде. В 2020 г. приблизительное число устройств, подключенных к интернету, выросло до 50 млрд. Это, помимо ПК и смартфонов, привычные нам кондиционеры, стиральные машины, холодильники, мультиварки и т. п.

А главным сдвигом стало то, что крупные корпорации начали воплощать вполне конкретные проекты в этой области.

Как устроен интернет вещей

Говоря простыми словами, интернет вещей — это совокупность вычислительных устройств, связанных между собой и способных принимать и передавать информацию по беспроводным сетям без вмешательства человека.

Интернет вещей — это не только устройства, но и датчики. Они взаимодействуют посредством облачного соединения. Информация, попав в облачное хранилище, сразу начинает обрабатываться программами. Затем принимается решение о том, какие действия нужно выполнить и в каком порядке (к примеру, настроить гаджеты и датчики так, чтобы не отправлялись уведомления, а пользователю не приходилось вводить данные вручную).

Если брать полный комплект систем IoT, то в него входят четыре отдельных компонента: пользовательский интерфейс, инструменты, отвечающие за обработку данных, способы подключения и датчики приборов.

Датчики приборов. Они предназначены для сбора информации в той или иной среде. Гаджет может быть оснащён целым рядом датчиков (таких как камера, акселерометр, GPS у смартфона). Они собирают в окружающей среде данные, чтобы с их помощью решать определённые задачи.

Способы подключения. Собранные сведения нужно как-то передать в облако. Можно сделать это разными путями: подключиться к интернету через спутник или Wi-Fi, воспользоваться LPWAN (энергоэффективными сетями дальнего радиуса действия), Bluetooth или старым добрым Ethernet. Какой вариант будет выбран, зависит от назначения того или иного гаджета в интернете вещей.

Инструменты обработки данных. Применяются уже в облачном хранилище, куда стекаются данные. Они подвергаются программной обработке для принятия решения о том, что делать дальше (например, перенастроить датчики, не заставляя пользователя вмешиваться, или отправить предупреждение). В некоторых случаях человек всё-таки должен сам ввести какие-либо данные, и тогда на помощь приходит пользовательский интерфейс.

Пользовательский интерфейс. Даёт возможность пользователю добавлять информацию или настройки в систему либо проверять её работоспособность. Тут последовательность циркуляции данных обратная: из интерфейса в облако и потом к датчикам, чтобы применить внесённые изменения.

Node-RED

Node-RED – это мощный инструмент для создания приложений IoT с уделением особого внимания упрощению соединения блоков кода для выполнения задач. Он использует подход визуального программирования,

который позволяет разработчикам подключать предопределенные кодовые блоки «узлы», вместе для выполнения задачи.

Связанные узлы, как правило, комбинация входных узлов, узлов обработки и выходных узлов, когда они соединены вместе, называют «потоки».

Как гласит первоисточник, Node-RED — это инструмент потокового программирования, первоначально разработанный командой IBM Emerging Technology Services и в настоящее время являющийся частью JS Foundation.

В качестве ключевой составляющей Node-RED выступает парадигма потокового программирования, которая была изобретена в 70-х Джеем Полом Моррисоном. Потоковое программирование — это способ описания поведения приложения в виде сети черных ящиков или «узлов», как они называются в Node-RED. Каждый узел имеет четкую цель — к нему поступают некоторые данные, он обрабатывает данные, а затем передает их на следующий узел. Сеть отвечает за поток данных между узлами.

Эта модель отлично подходит для того, чтобы представить ее визуально: так она становится более доступной для широкого круга пользователей. Если кто-то пытается разобраться в проблеме, он может разбить задачу на отдельные шаги, взглянуть на поток и понять, что он делает, без необходимости разбираться в отдельных строках кода в каждом узле.

Node-RED работает в среде исполнения Node.js, а для создания или редактирования потока («Flow») используется браузер. В браузере вы можете создавать свое приложение путем перетаскивания необходимых узлов («Node») из палитры в рабочую область и соединять их вместе. Одним кликом по кнопке «Deploy» приложение разворачивается в среде исполнения и запускается.

Палитра узлов может быть легко расширена путем установки новых узлов, созданных сообществом, а созданные вами потоки могут быть легко переданы в виде файлов JSON.

Node-RED начал свою жизнь в начале 2013 года как совместный проект Ника О'Лири и Дэйва Конвея-Джонса из группы IBM Emerging Technology Services.

Почему проект называется Node-RED?

Со слов авторов: «Название было веселой игрой слов, звучащих как «Code Red». Часть «Node» отражает суть модели потокового программирования (поток/узел) и основную среду выполнения Node.JS. Окончательного решения о том, что же означает часть «RED», принято так и не было. Одно из предложений — «Rapid Event Developer» (быстрый разработчик событий), но мы никогда не чувствовали себя обязанными что-либо формализовать».

Node-RED предоставляется по лицензии Apache 2.0. Лицензия разрешает коммерческое использование, но при этом накладывает и ряд ограничений. Вот основные из них: при использовании торговой марки Node-RED (принадлежащей OpenJS Foundation) нельзя ее искажать; кроме того, есть ограничение ответственности (Liability/Warranty) — участники проекта не могут быть привлечены к ответственности в случае причинения убытков в процессе некорректного использования их продукта.

Практическая работа 1. Установка Node-red и палитр компонентов. Экспорт и импорт проекта

Цель: познакомиться с назначением и вариантами установки Node-red, научиться устанавливать программу в соответствии с целями применения.

Первый вариант установки позволяет выполнять все функции, кроме управления роботами через локальный сервер. Второй вариант установки с развёртыванием сервера на основе операционной системы Ubuntu предназначен для командной работы по управлению роботами в компетенции «Интернет вещей» на чемпионате Профессионалы.

Порядок действий (1 вариант)

1. Установить node.js

Установку можно осуществить на разных операционных системах. Для установки нужно скачать актуальную версию. если установка осуществляется на компьютер с операционной системой Windows, то нужно действовать через командую строку или Windows Power Shell. Разработчики node.js дают скрипт для установки.

installs fnm (Fast Node Manager)

winget install Schniz.fnm

configure fnm environment

fnm env --use-on-cd | Out-String | Invoke-Expression

download and install Node.js

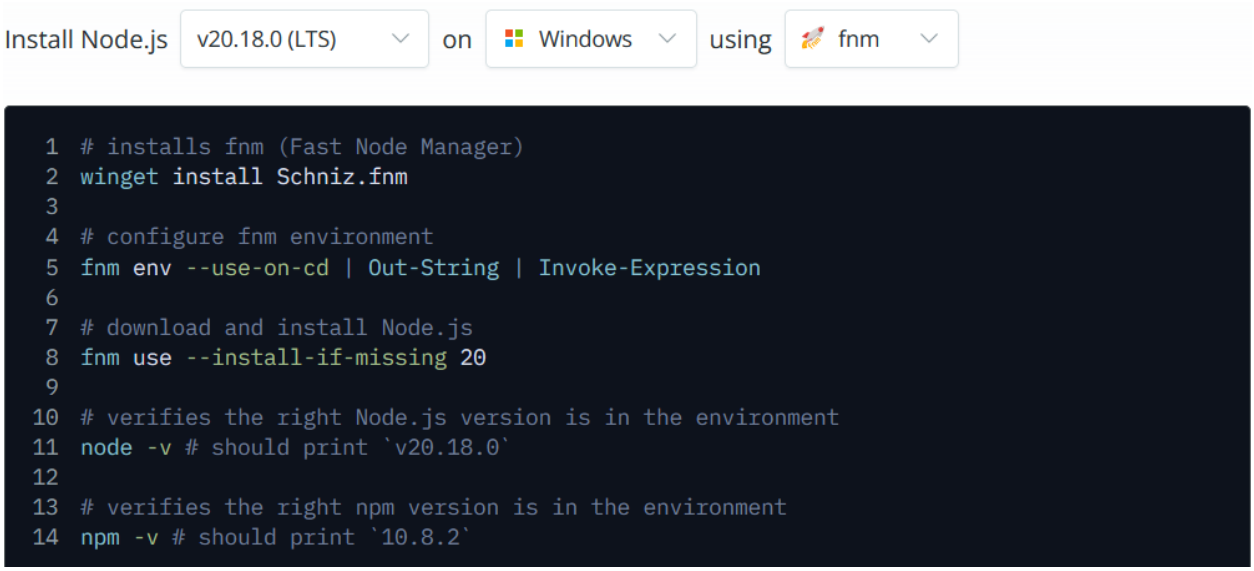
fnm use --install-if-missing 20

verifies the right Node.js version is in the environment

node -v # should print `v20.18.0`

verifies the right npm version is in the environment

npm -v # should print `10.8.2`



2. Затем, нужно установить саму сетевую версию Node-red. Предварительно не забудьте разрешить Power Shell выполнять сценарии.

npm install -g --unsafe-perm node-red

3. Запуск осуществляется командой:

node-red

4. После запуска в браузере наберите адрес http://127.0.0.1:1880/

Порядок действий (2 вариант для роботов)

Необходимое программное обеспечение:

– программа для создания виртуальных машин Oracle VirtualBox;

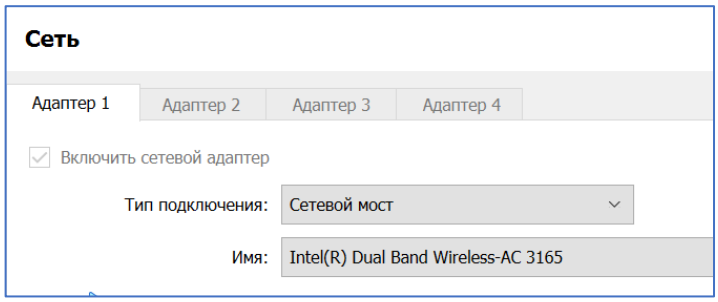
– Ubuntu_Node-red - v1.0.3 (или другая актуальная версия).

1. Установите Oracle VirtualBox (если не установлено).

2. Скачайте с сайта производителя образ, который будет использоваться вместе с virtualbox. Образ называется Ubuntu_Node-red - v1.0.3, либо другой актуальной версии.

3. Запустите Ubuntu_Node-red - v1.0.3, при этом будет создана виртуальная машина с заданными параметрами.

4. Поменяйте сетевые параметры. В настройках виртуальной машины в разделе "Сеть" необходимо выставить "Тип подключения: Сетевой мост



5. Запустите виртуальную машину. Логин и пароль для входа в систему redadmin, 12345.

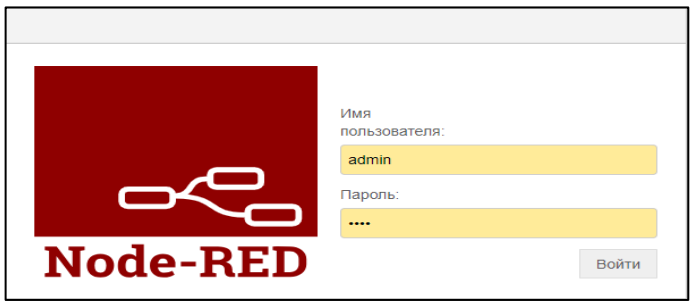
6. После загрузки операционной системы определите IP-адрес виртуальной машины.

7. Наберите в любом браузере полученный адрес и номер порта 1880. В данном случае 192.168.1.194:1880.

8. Выполните настройку аутентификации

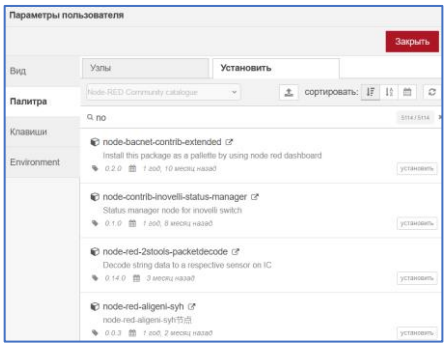
В версии v1.0.3, установлен логин и пароль на Node-red.

По умолчанию логин:пароль => admin:1234



Программу Node-red можно настроить на русский язык (Параметры пользователя), но перевод частичный. Также есть возможность добавлять ноды.

Установка дополнительных модулей



Установите модуль node-red-dashboard:

1. Зайдите в веб-интерфейс Node-RED.
2. В правом верхнем углу вызовите меню.
3. В открывшемся окне выберите вкладку **Управление палитрой** (Manager palette).
4. Перейдите на вкладку **Установка** (Install), введите в поле поиска **dashboard** и нажмите на клавиатуре Enter.
5. Установите пакет с названием **node-red-dashboard**.
6. Закройте окно с настройками. Установка модуля завершена.

Принцип работы в Node-red

Выбрать нужные ноды, перетащить, соединить (drag&drop), настроить каждую ноду при необходимости, сохранить (Deploy, Развернуть). При необходимости дописываем функции (на языке Java). Выполняем отладку. Создаём интерфейс пользователя. Всё это в следующих практических работах.

Сохранить проект можно в файле формата json. Для этого в меню нужно найти команду Экспорт. Готовый файл можно импортировать.

Практическая работа 2. Базовые ноды Node-red: умный дом

Цель: познакомиться с основными нодами Node-red. Научиться их применять для автоматизации процессов.

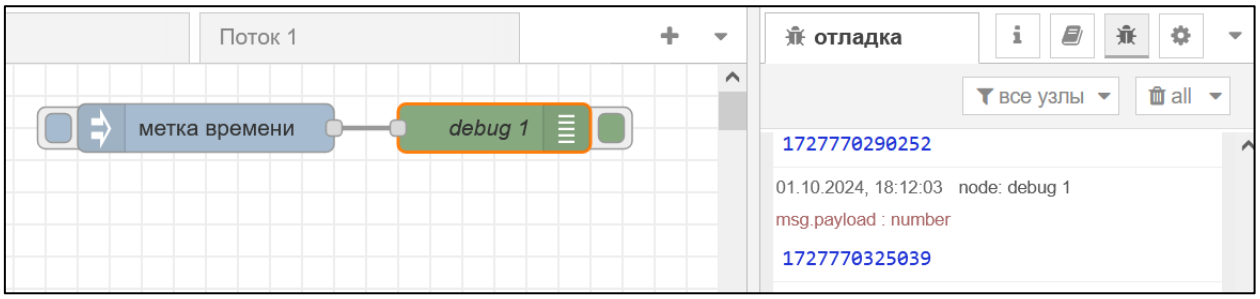
План работы

1. Ноды inject и debug.
2. Нода switch.
3. Нода change.
4. Trigger, delay.
5. Link-ноды.
6. Общий пример.

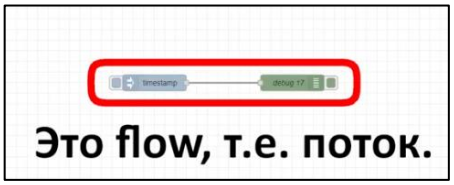
Ход работы

1. Начнём с двух базовых нод, которыми вы будете пользоваться чаще всего. Это ноды inject и debug.

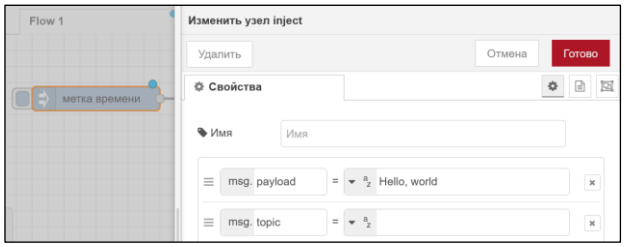
Соединяем ноды, выполняем команду Развернуть, сохраняем конфигурацию. При переходе на значок жучка (Отладочные сообщения), видим, что при нажатии на Метку времени (inject) появляются какие-то числа, и при каждом дополнительном нажатии на квадратик число на несколько единиц увеличивается. Это не что иное, как число секунд, прошедших с 1 января 1970 года. По умолчанию нода inject всегда выводит эту временную метку.



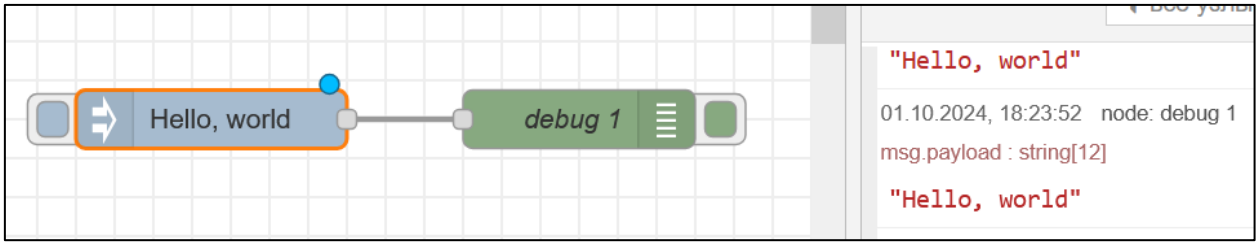
Нода debug получает сообщение и выводит их на вкладку отладки. Если бы ноды не были связаны, то метка времени посылала бы информацию в никуда (на деревню дедушке). Взаимосвязанные ноды – это flow, поток. Две связанные ноды, это уже небольшой, но поток.



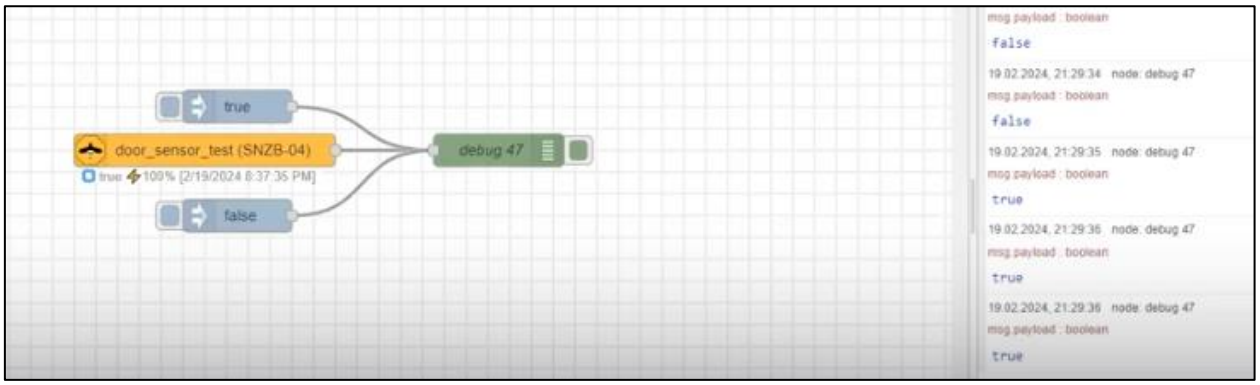
Если мы хотим сделать классический для программирования Hello, world! то выглядеть он будет вот так:



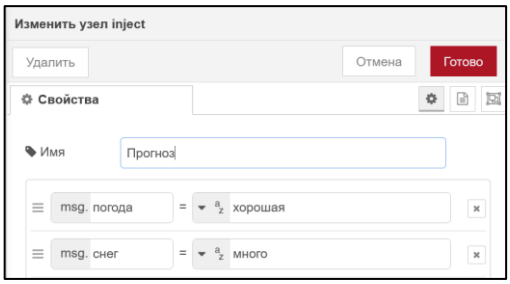
Окно изменения узла открывается двойным щелчком.



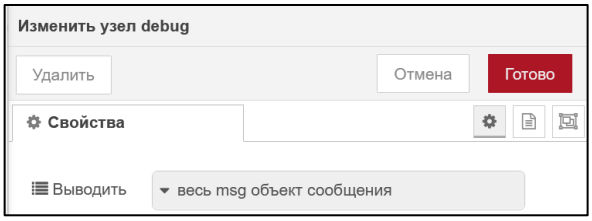
Одно из самых популярных использований ноды inject симуляция сообщений от какого-либо устройства. Допустим, вы делаете автоматизацию на основе датчика открытия двери, но, чтобы проверять программу, мы будем не бегать открывать закрывать дверь, а имитировать открытие с помощью ноды inject.



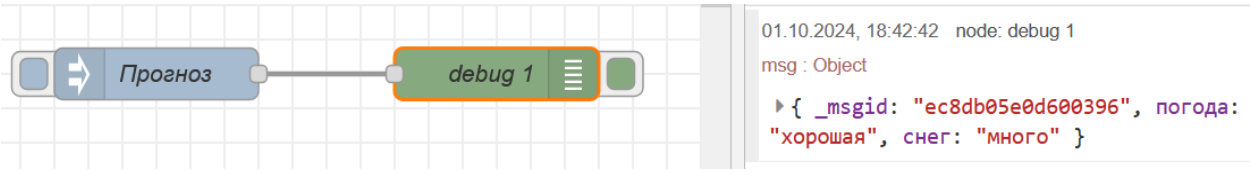
Для следующего примера соберите поток из инжект и дебаг. Метка времени должна передавать в ключе Погода значение Хорошая, а в ключе Снег – значение Много.



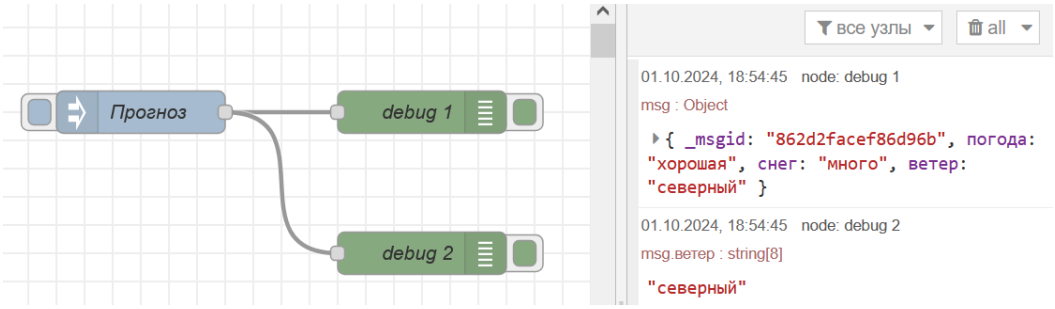
В настройках дебаг мы укажем, что ходим видеть весь объект, который приходит в данную ноду, а не только то значение, которое было получено в ключе payload.



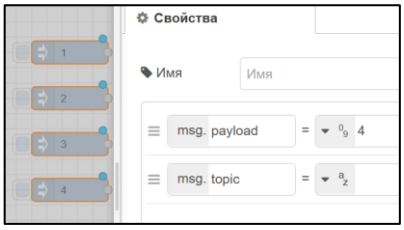
И оказывается, то Node-red передаёт не простые сообщения, а java-объекты, которые могут содержать множество свойств. Свойством называется пара ключ-значение



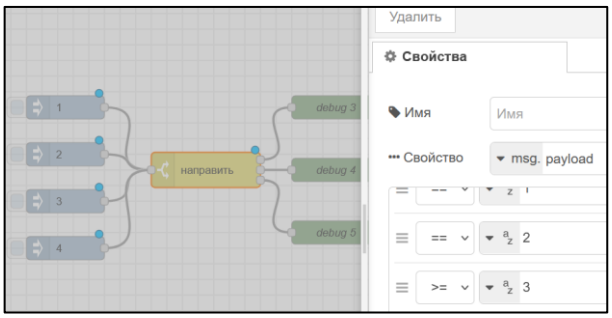
В дебаге можно вывести весь объект, а можно только определённую пару ключ-значение. Например, первый дебаг оставим без изменения, а во второй выведем свойство объекта – ветер.



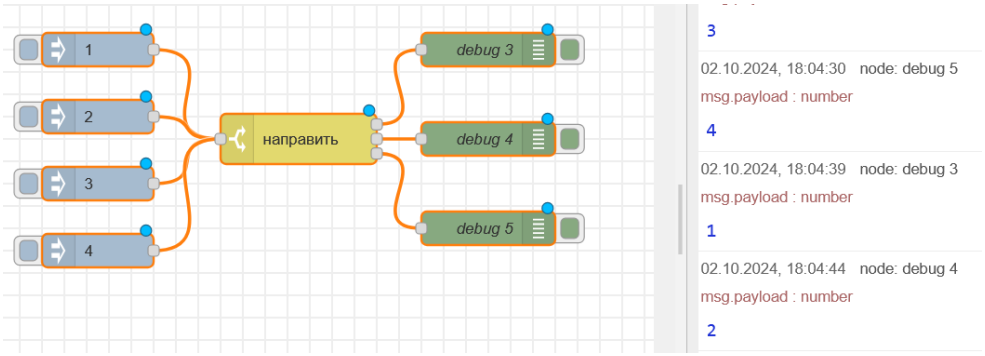
2. Нода switch – условная конструкция. Для примера создайте 4 ноды инжект, которые будут передавать в payload значения 1, 2, 3, 4.



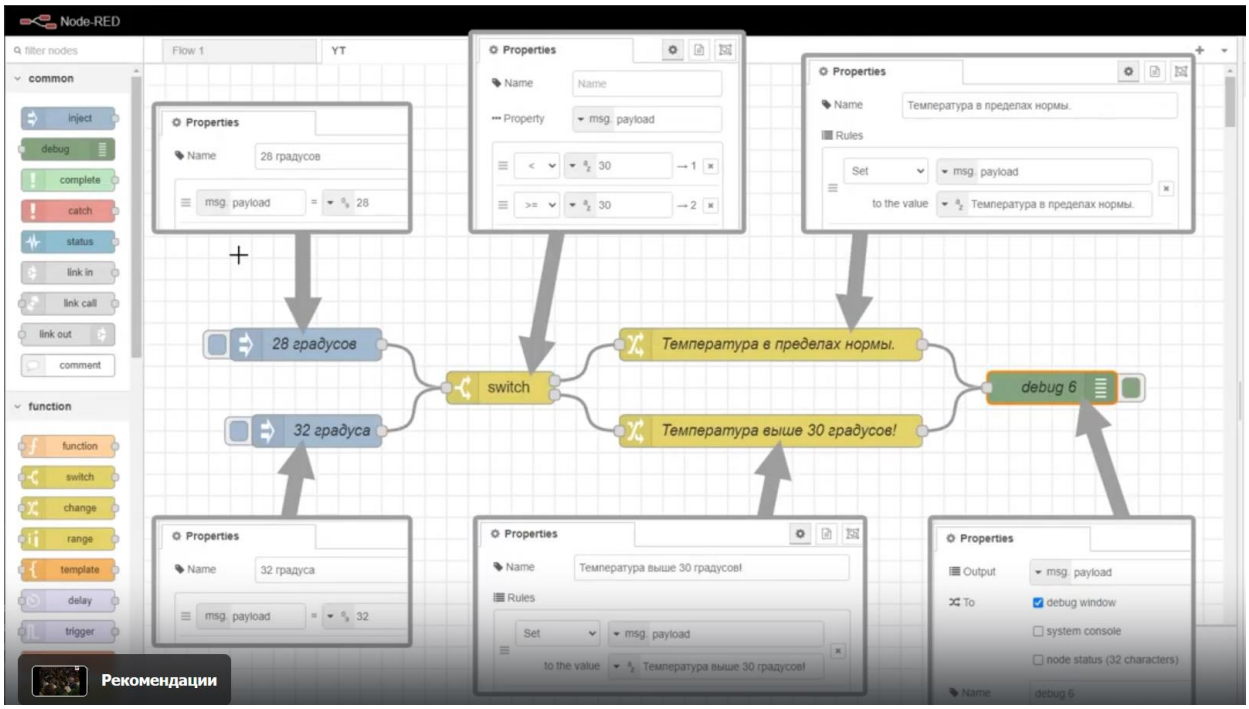
Добавьте ноду свитч, которая также будет смотреть за значением payload. Если нода получит сообщение 1, то полученное сообщение появится из верхней точки выхода, если 2, то из средней, если больше либо равно трём, то сообщение будет направлено на нижнюю точку выхода. К каждому выходу нужно добавит дебаг.



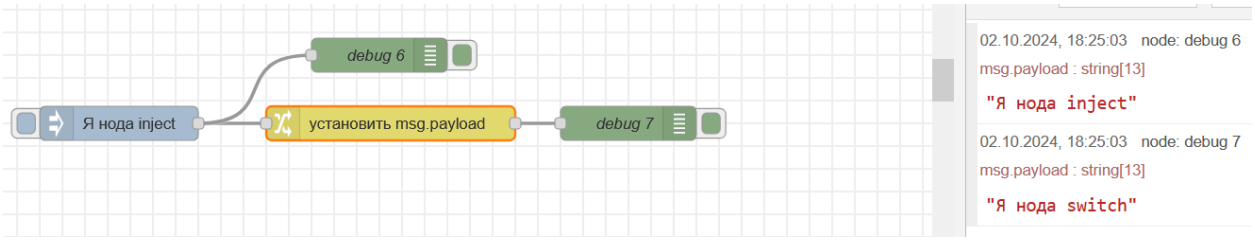
Проверьте, что нода свитч рассортировала сообщения, согласно правилам настройки.



У ноды свитч может быть огромное количество сценариев, например, мониторинг температуры. Система отправляет сообщения с какого-либо датчика

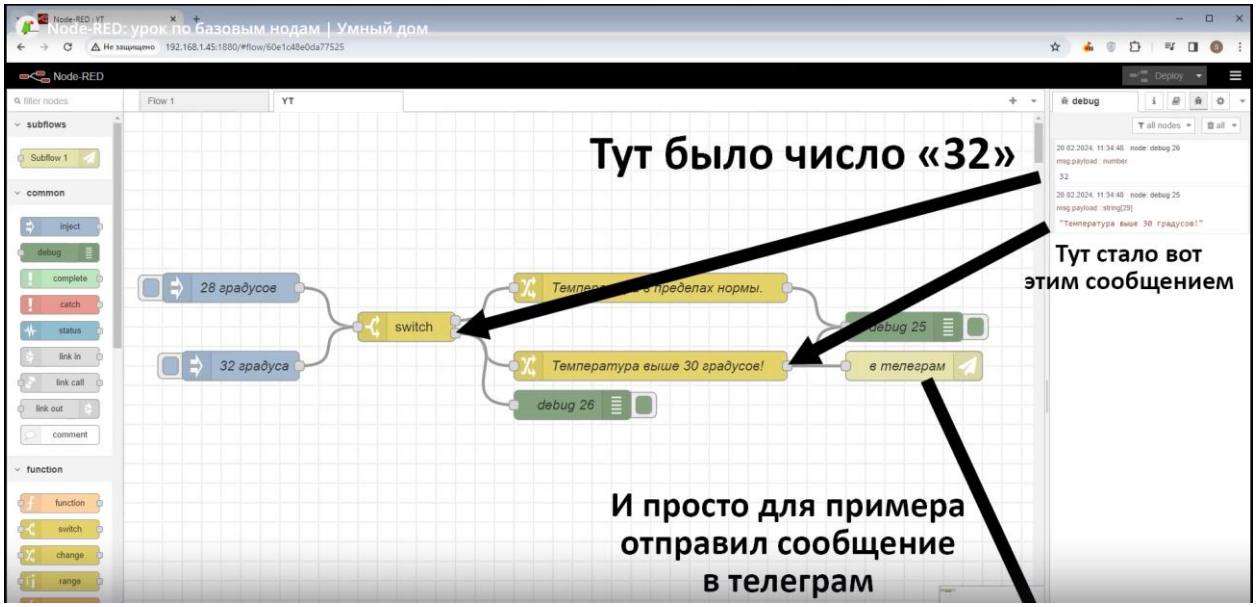


3. Далее идёт нода change, которая может создавать и изменять переменные. Для примера задайте значение метке времени «Я нода inject», а в свитче поменяете его на «Я нода switch», поставьте два дебага и проверьте отладочные сообщения.

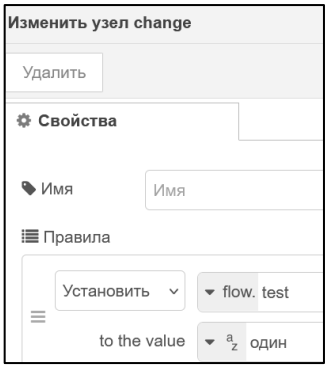


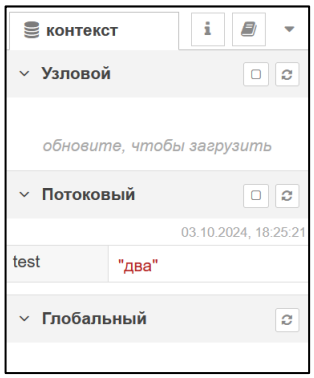
В качестве примера использования данной ноды можно взять прошлый пример с нодой свитч, в котором меняется число, имитирующее температуру.

В свитч приходит число, а выходит сообщение. В Телеграмм уходит уведомление о превышении температуры.



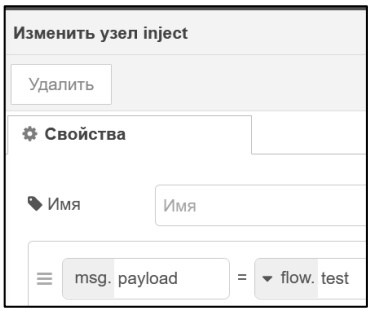
У ноды ченч есть ещё одна важная функция. Она позволяет создавать переменные контекста или ключи контекста. Переменные можно создавать в контексте flow - доступные в одной вкладке и в контексте global, доступные во всех вкладках. Установите переменную flow.test в значение один и в значение два. Переменную можно посмотреть.



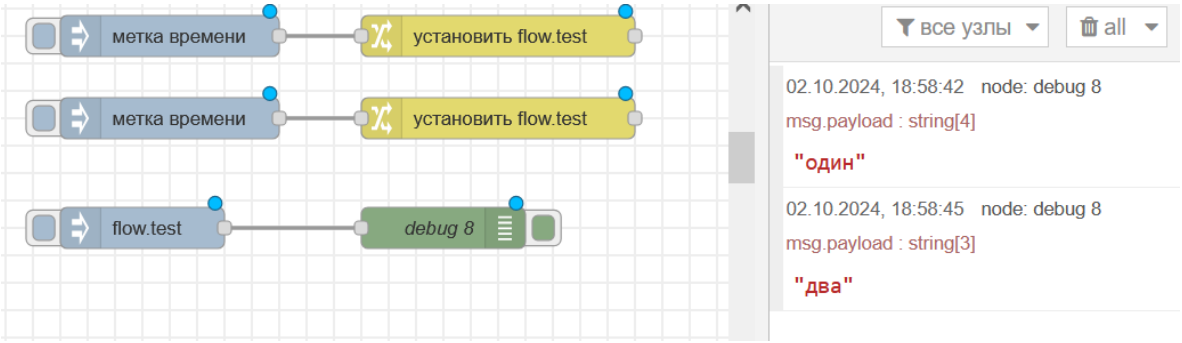


Для просмотра переменной нужно открыть контекстные данные, в окне которых выводятся значения потоковых и глобальных переменных.

Для того, чтобы увидеть значение переменной в отладочном сообщении создайте связку инжект-дебаг и отправьте переменную в свойство payload.

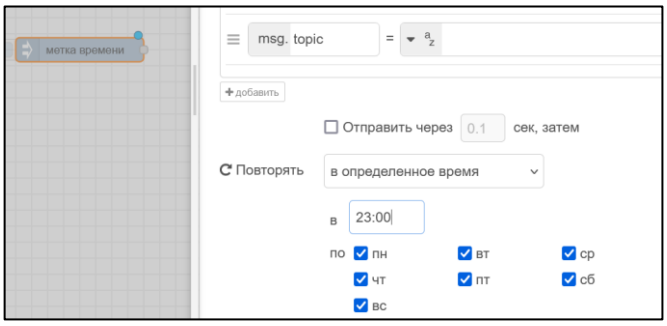


Меняйте значение переменной, щёлкая по квадратам инжектов, выводите на экран. В панели дебаг будет выводиться то, что сейчас находится в переменной.



Переменные контекста необходимы, когда по той или иной причине нужно в разное время и, возможно из разных мест обращаться к одному и тому же значению, к примеру, у вас на двери находится датчик открытия и на Алису присылается сообщение, но вам не нужно, чтобы это сообщение присылалось с 23-00 до 7-00, если вы пришли в это время. Чтобы Алиса могла говорить только днём, нужно при помощи всё тех же нод inject задать допустимое для разговора время.

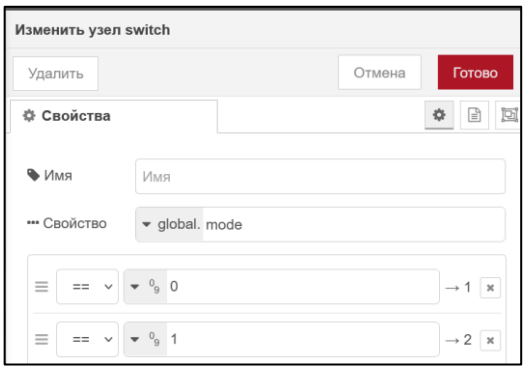
Создадим ноды, которые будут включать и выключать «тихий режим» в ночное время. Для задания времени нужно изменить свойство Повторять.



Для ноды ченч в контексте global в имя переменной mode установим логическое значение false. Нужно скопировать ноду и поменять присваемое

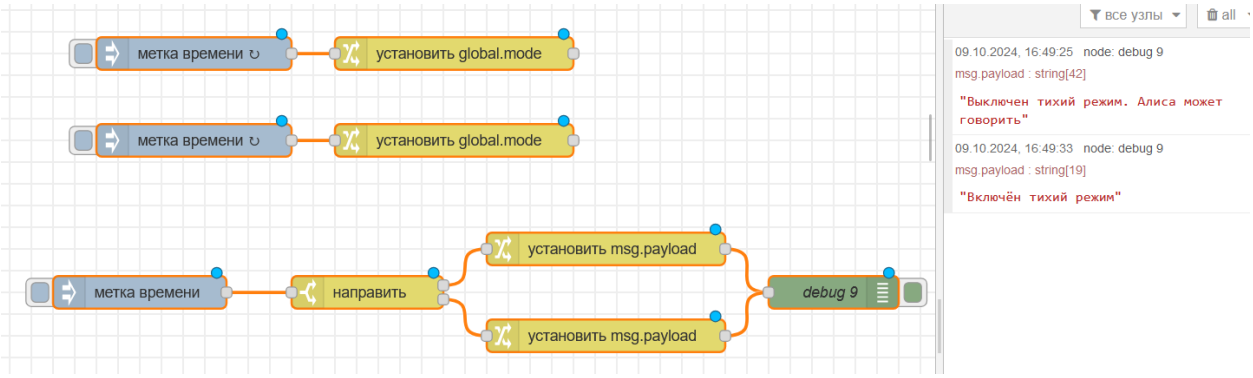
значение на true. Можно просмотреть значение глобальной переменной в окне контекстных данных.

Остаётся создать простое выражение if…else при помощи ноды свитч, которая будет направлять сообщение на одну из двух точек выхода. Нужно поменять свойство на global.mode, так как мы хотим проверять именно эту переменную контекста, а не payload входящего сообщения.



При проверке условия нельзя сравнить с true и false (теми значениями, которые мы задавали в ноде ченч), но 1 и 0 – это тоже самое, но в числовом варианте.

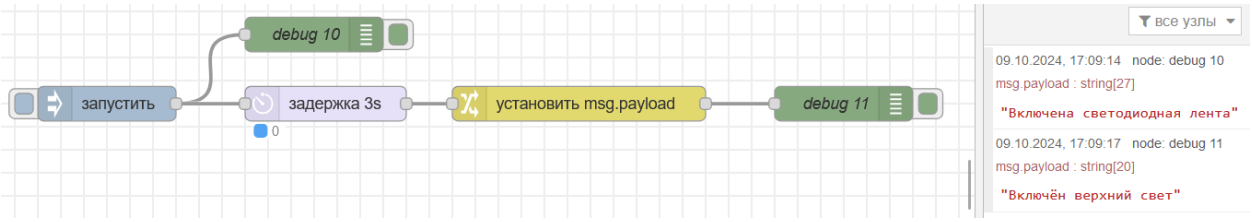
Теперь создадим пару нод change с сообщениями («Тихий режим включён», «Тихий режим выключен. Алиса, можешь говорить») и, чтобы это всё заработало добавляем inject и debug. Вручную переключая режим времени и генерируя сигнал, получаем разные значения сообщений для разных временных промежутков.



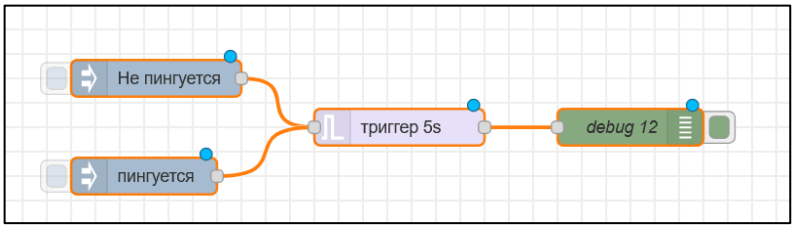
Таким образом, данная автоматизация будет вечером включать тихий режим, а утром выключать, чтобы Алиса могла оповещать вас о каких-то событиях, но только днём. Если эти события случаются в ночное время, то Алиса будить вас не будет.

4. При создании тех или иных автоматизаций периодически нужно выполнять действия с паузами между ними. Решить эту задачу поможет нода delay.

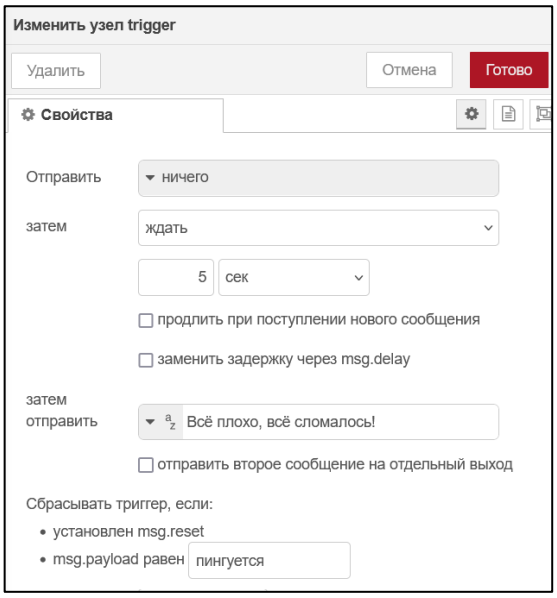
Представим комнату. Допустим при нажатии выключателя света должна сразу загораться светодиодная подсветка, а через три секунды верхний свет. Сделать такую автоматизацию можно следующим образом.



Также в Node-red можно создавать автоматизации с помощью триггеров. Например, мы пингуем какое-либо устройство и как только сигнал перестаём поступать, вы должны получить сообщение о недоступности данного устройства. Но мы не хотим получать сообщение сразу, вдруг это был секундный сбой. Нужно как-то отфильтровать короткие перебои и с этим нам поможет нода trigger.



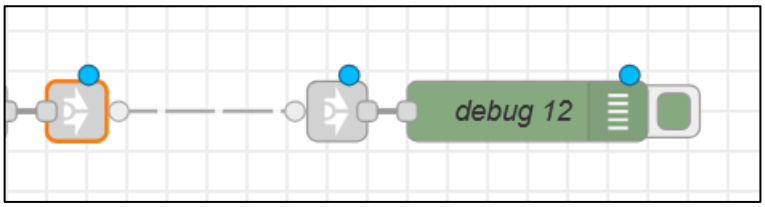
По умолчанию, если ничего не происходит нода триггер отправляет 1, но нам этого не нужно, поэтому в настройке Отправить указываем Ничего. Далее указывает, сколько нода будет ждать, прежде чем оповестит нас об отсутствии пинга (5 секунд). Далее нужно написать, что именно будет отправлено из ноды. В поле «Сбрасывать триггер» нужно указать, что будет останавливать триггер от отправки данного сообщения. Пишем то, что написано во втором inject, то есть слово «пингуется».



Если при проверке мы нажмём «Не пингуется», то запустится таймер. Если в течение 5 секунд ничего не изменится, то выведется сообщение. Если же за 5 секунд мы нажмём ноду «пингуется», то триггер сбросится и сообщение не поступит. То есть получается, что пинг прервался и запустил работу триггера, но потом возобновился и необходимость в сообщении исчезла.

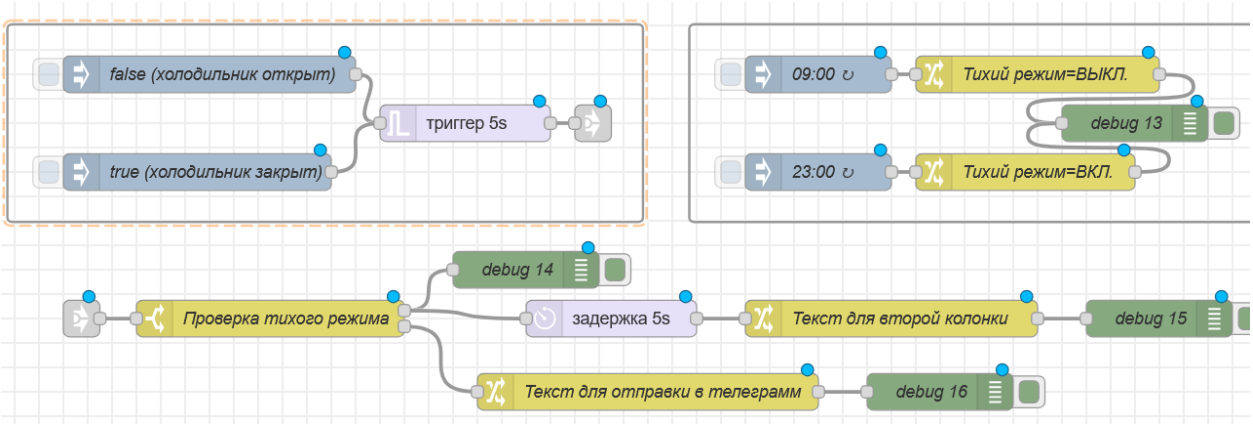
5. Link-ноды на старте обучения, возможно, и не нужны, они не самые важные, но без них в какой-то момент нельзя буде обойтись. Рано или поздно наступит момент, когда в вашей автоматизации нужно будет подключить ноду, находящуюся справа, к ноде, которая располагается сильно левее. Сделать это обычной связью можно, но это будет выглядеть некрасиво. Плюс, если таких связей будет много, то читаемость потока будет нулевой. Также эти ноды умеют соединять потоки на разных вкладках. Работать с ними можно, как с обычными нодами, вытаскивать и соединять. Единственное отличие, что, когда они не выделены, связи не видно.

Второй вариант создания такой связи: нужно выбрать обычное соединение, из контекстного меню выбрать Insert, Link Nodes.



Таким образом мы можем разнести ноды по различным областям и вкладкам, и поток будет работать без проблем.

Задача. Если дверь воображаемого холодильника будет открыта слишком долго и сейчас не ночь, то мы сначала получим уведомление на одну Яндекс-станцию, а затем на вторую, но, если сейчас ночь, то мы просто получим уведомление на Телеграмм, при этом колонки будут молчать. При этом, если дверь будет закрыта в течение 5 секунд и сработает триггер, что ни на станции, ни в Телеграмм никаких сообщений подаваться не будет.



Настройте ноды, укажите параметры, протестируйте поток.

Практическая работа 3. Создание Dashboard

Цель: научиться создавать в Node-RED интерактивные панели (dashboards) с помощью набора виджетов.

Подготовка

Для создания интерактивной SVG-панели на контроллере Wiren Board с помощью Node-RED нам понадобится:

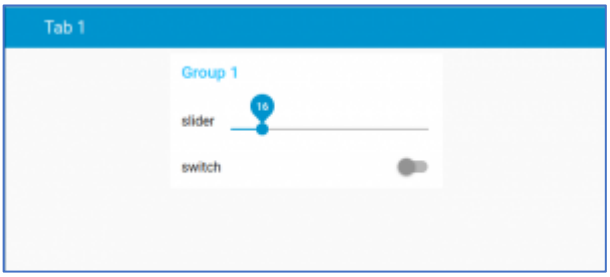
– модули node-red-dashboard и node-red-contrib-ui-svg.

Концепция панелей в Node-RED

В Node-RED панель состоит из *Вкладок* (Tabs), которые содержат *Группы* (Groups). Группы в свою очередь могут содержать готовые или пользовательские *Виджеты* (Widgets), или *svg-изображения*. Ограничения на количество вкладок и групп в документации не описаны.

Между табами можно переключаться из меню, или с помощью ноды *ui control*. Если вы используете устройство с маленьким экраном, то меню можно отключить для экономии места.

Создание панели с виджетами

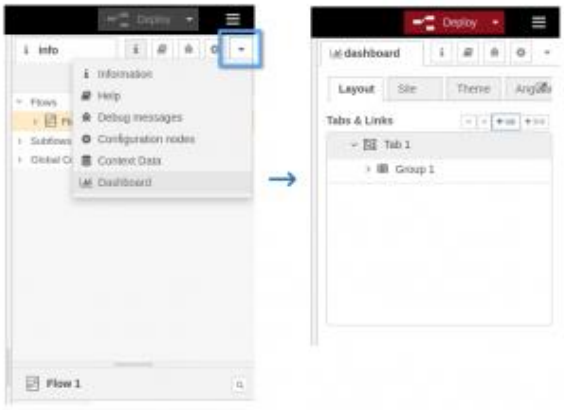


Панель с виджетами

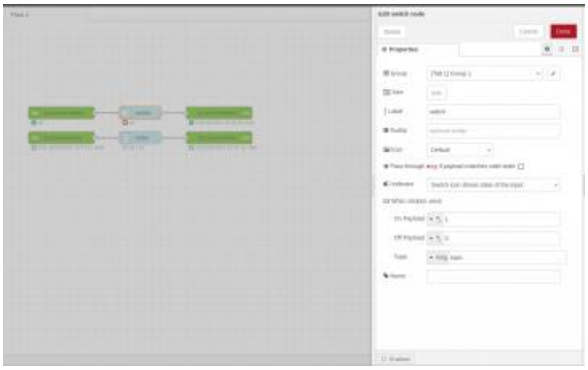
Панель с типовыми виджетами — это быстрый способ получить несложный интерфейс, который закроет большинство потребностей. Притом, интерфейс получается адаптивный, то есть при нехватке места по горизонтали, группы располагаются друг под другом.

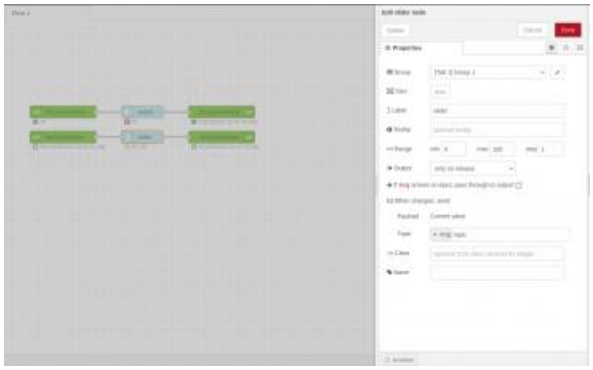
Создадим панель, на которую поместим элементы управления зуммером контроллера:

1. Перейдите в веб-интерфейс Node-RED.
2. В правом верхнем углу нажмите на треугольничек и выберите инструмент **Dashboard**.
3. В **Layout** добавьте новую вкладку, а в ней — группу.



4. В левой панели в группе **dashboard** найдите ноды **switch** и **slider** и перетащите их в рабочую область.
5. По очереди дважды кликните на добавленных нодах, проверьте, что выбрана верная группа *Group 1*:
6. В настройках ноды **switch**:
    - снимите флажок **Pass through msg if payload matches valid state**;
    - в поле **Indicator** выберите **Switch icon shows state of the input**;
    - В полях **On Payload** — a/z: **1**, **Off Payload** — a/z: **0**.





7. В настройках ноды **slider**:
    - снимите флажок **Pass through msg if payload matches valid state**;
    - в поле **Range** установите значение **max = 100**;

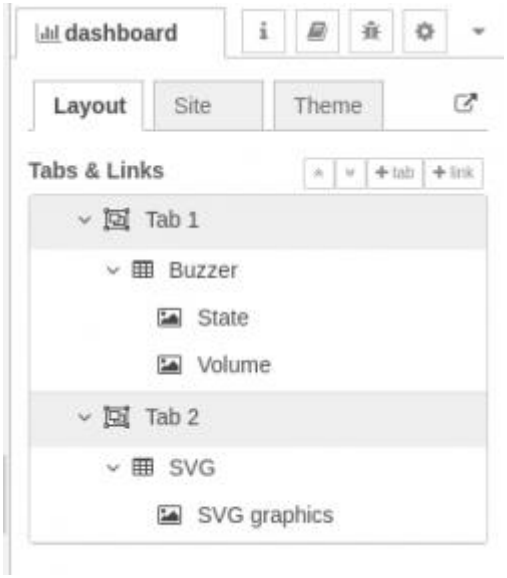
- в поле **Output** выберите **only release**.

8. Добавьте два экземпляра inject на панель.
9. Добавьте два экземпляра debug на панель.
10. Соедините ноды между собой по картинке ниже и нажмите вверху кнопку **Deploy**.

Теперь в инструменте **dashboard → Layout** нажмите на кнопку со стрелочкой, откроется новая страница с нашей панелью.

Создание svg-панели

Добавление вкладки



Переименованные элементы

Добавим новую вкладку и создадим в ней svg-панель:

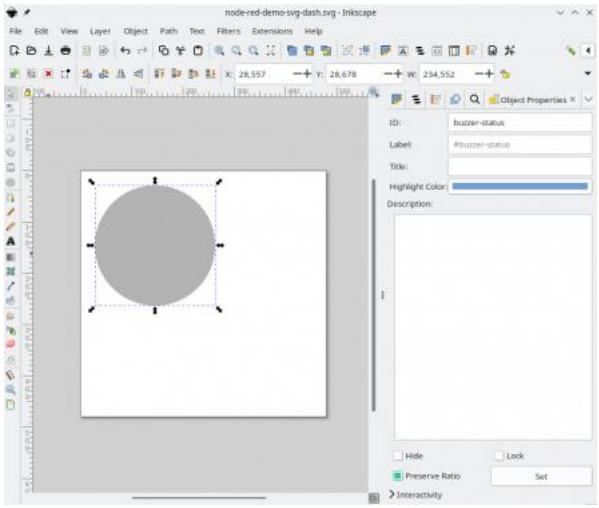
1. Чтобы было удобнее, переименуйте элементы созданной выше панели:
   - Group 1 → Buzzer
   - slider → Volume
   - switch → State
2. Добавьте новую вкладку **Tab 2** и группу **SVG**.
3. В настройках группы укажите ширину 9.
4. В левой панели в группе **dashboard** найдите ноду **SVG graphics** и перетащите её в рабочую область.
5. Дважды кликнете на добавленной ноде и настройте её:
   - в поле **Group** выберите **[Tab 2]SVG**.
   - на вкладке **SVG** надо добавить код svg-изображения.

Создание изображения

В ноде *SVG graphics* есть демо-версия встроенного svg-редактора, но мы будем использовать популярный *Inkscape*.

Создадим изображение в Inkscape:

1. Откройте редактор и меню **File → Document Properties**.
2. Установите размер страницы, например, 480 x 480 px.
3. Добавьте серый круг, выберите его и откройте панель с параметрами объекта, меню **Object → Object Properties**. Справа откроется дополнительная панель.
4. В панели **Object Properties** в поле **ID** напишите имя элемента: buzzer-status. Это имя мы будем использовать при фильтрации событий и обращении к его свойствам.

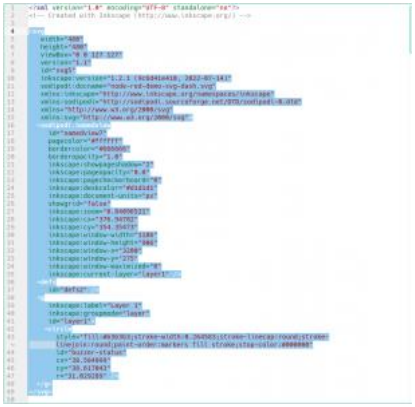


5. Аналогичным образом добавьте другие элементы, например, текст и квадрат.
6. Сохраните файл в формате SVG.

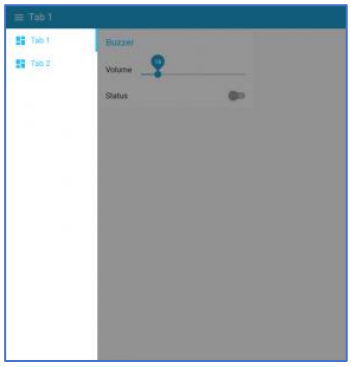
Импорт изображения

Функции импорта в ноде *SVG graphics* нет, поэтому мы просто скопируем содержимое нашего svg-файла и вставим его в специальное поле ноды:

1. Откройте созданный выше svg-файл в текстовом редакторе и скопируйте содержимое вместе с тегами <svg>.
2. Перейдите в настройки ноды **SVG graphics** и откройте вкладку **SVG**.
3. Удалите весь текст и вставьте скопированное раньше содержимое svg-файла.
4. Нажмите кнопку **Done** и **Deploy**.

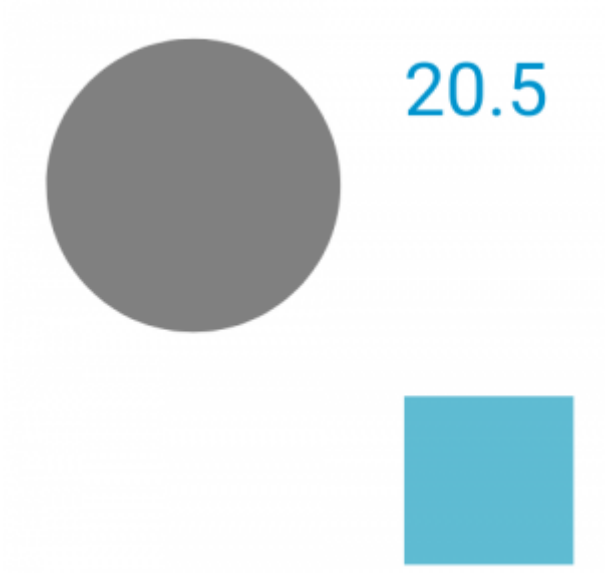


Снова перейдите на страницу с панелью, теперь там должно появиться меню, кликните на него и выберите **Tab 2**.



Описание флоу

Пример флоу SVG-панели



Пример SVG-панели

Пример флоу и svg-файл файл можно скачать на GitHub node-red-demo-svg-dashboard. В примере используются только системные устройства, поэтому он должен работать на всех контроллерах Wiren Board без доработок.

Панель в примере состоит из двух вкладок:

1. **Tab 1** вкладка с виджетами:
    - State — переключатель, который отображает состояние зуммера и управляет им.
    - Volume — ползунок, который показывает и изменяет громкость зуммера.
    - SVG Dash — кнопка, которая открывает вкладку **Tab 2**.
2. **Tab 2** — вкладка с svg-изображением, которое хранится в ноде **SVG graphics**. Элементы изображения:
    - Серый круг *#buzzer-state* — кнопка, которая отображает и переключает состояние зуммера.
    - Текстовое поле *#temperature* — отображает температуру процессора. Чтобы изменять текст, надо обратиться к селектору *#temperature > tspan*.
    - Голубой квардрат *#buzzer-control* — кнопка, которая открывает вкладку **Tab 1**.

Нода **ui control** переключает вкладки панели и принимает на вход имя вкладки, которую нужно открыть.

В ноде **SVG graphics** на вкладке **Event** описаны события, которые будут генерировать элементы *#buzzer-state* и *#buzzer-control*. В обоих случаях на выход ноды **SVG graphics** будет отправлено сообщение с информацией о событии и элементе, вызвавшим его. Сообщения с описанием событий обрабатываются нодой **switch-events**.

На вкладке **Binding** ноды **SVG graphics** описаны свойства, которые можно менять с помощью соответствующих сообщений: у элемента *#buzzer-state* будет меняться свойство **fill**, а в элементе *#temperature > tspan* будет

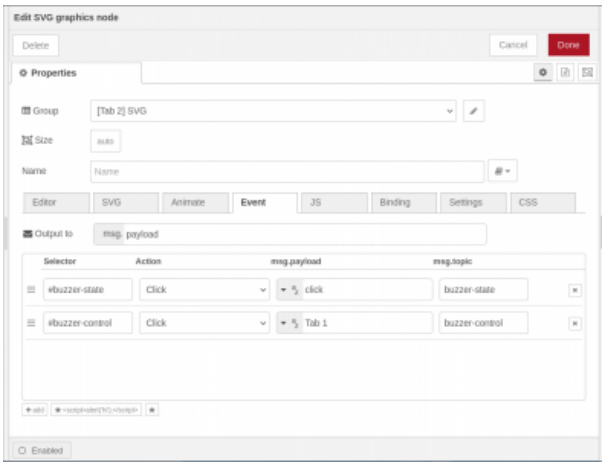
изменяться текст. Сообщения для изменения свойств генерируются нодами **set-color** и **set-temperature**, которые содержат примерно такой код:

```
return {
    "payload": {
        "buzzer": {
            "color": (msg.payload == '1')? 'yellow': 'gray'
        }
    },
    "topic": "databind"
};
```

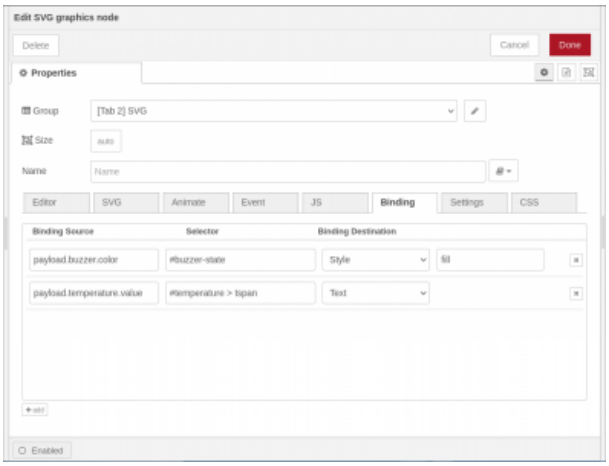
Нода **invert-state** инвертирует состояние зуммера.

Нода **change-tab** генерирует событие при переключении вкладок панели и обновляет значение температуры процессора в текстовом поле svg-панели.

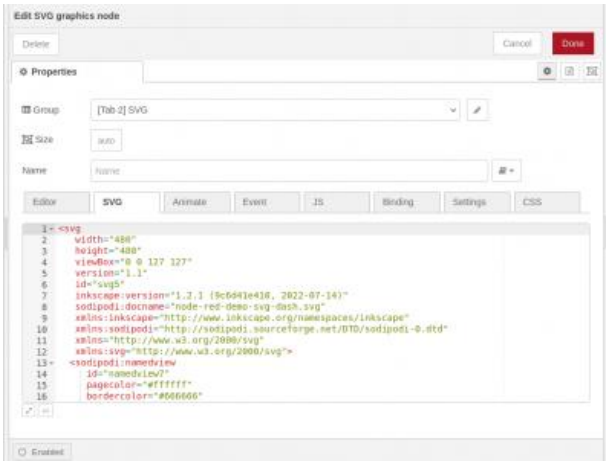
Настройки ноды *SVG graphics*



События

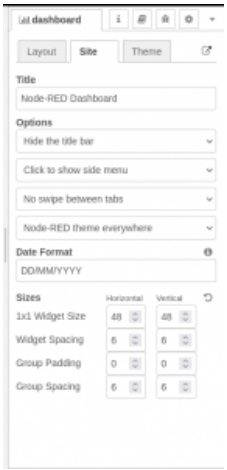


Привязки



Код SVG

Хитрости



Верхнюю панель можно скрыть: перейдите **dashboard → Site** и выберите **Hide the title bar**. Здесь же находятся настройки меню и размеров виджетов.

Тему можно изменить: перейдите **dashboard → Theme** и выберите одну из тем.

Скрыть заголовок группы можно в настройках, параметр **Display group name**.

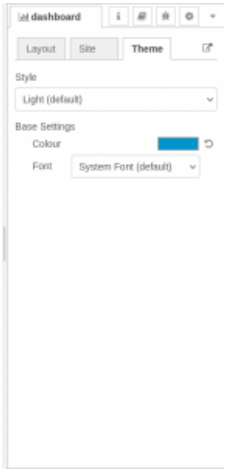
Чтобы скрыть однопиксельную рамку вокруг svg-панели, добавьте на вкладку **SVG** ноды **SVG graphics** перед тегом <svg> код:

```
<style>
   body.nr-dashboard-theme md-toolbar {
      background-color: #1a1a1a;
      color: #fff;
   }
   md-card.nr-dashboard-template {
      padding: 0px;
   }
   ui-card-panel.visible {
      top: 0 !important;
      left: 0 !important;
```

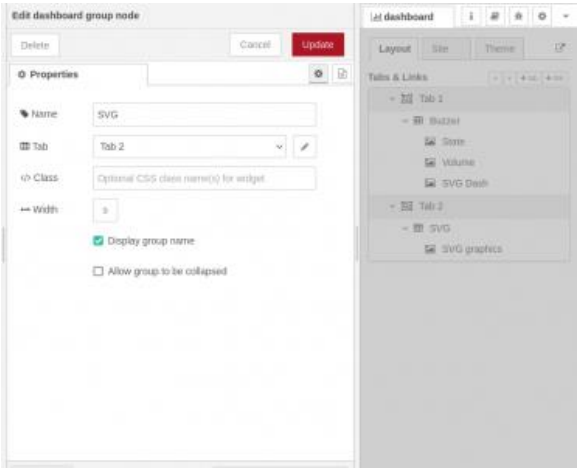
```
    border: none;
  }
</style>
```

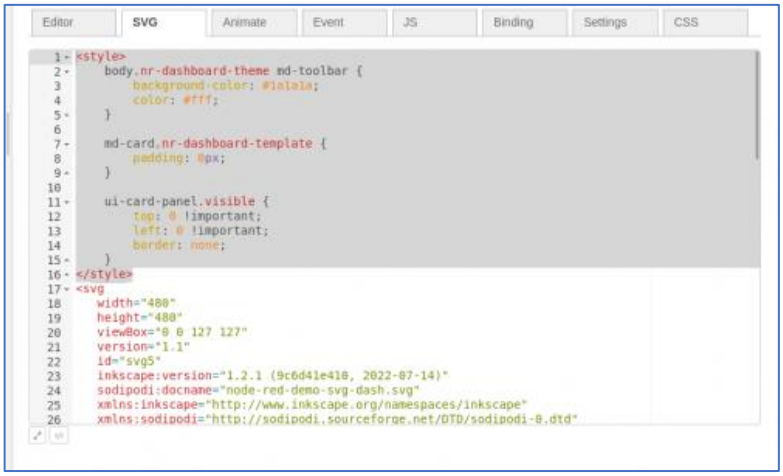
- Хитрости

  Видимость панели



Темы





Практическая работа 4. Использование Node-red в промышленном Интернете вещей

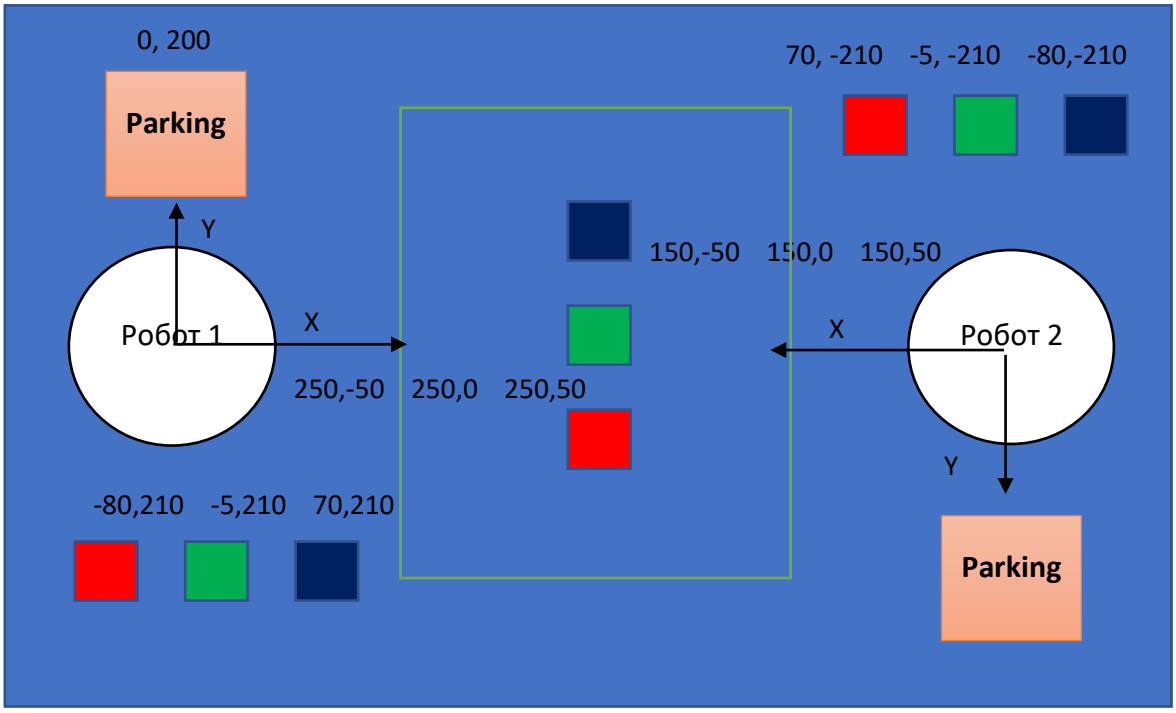
Цель: получить опыт управления промышленным роботом и светофором при автоматизации производственного процесса. Познакомиться с устройствами, направлением задания и реализацией, которые нужны при подготовке к чемпионату «Профессионалы» по компетенции «Интернет вещей». Узнать возможности среды Node-red в этой области.

План:

1. Постановка задачи.
2. Знакомство с оборудованием.
3. Применение Node-Red.
4. Создание интерфейса пользователя.

Ход работы

1. Постановка задачи



С помощью роботов-манипуляторов организовать перемещение объектов (кубиков) на заданные позиции (имитация сборки устройства).

Учитывайте, что точка начала координат для каждого робота – это центр опоры.

Создать интерфейсы оператора и инженера. Добавить управление светофором (включение по нажатию кнопки). Добавить возможность парковки.

Создайте интерфейс оператора. Организуйте управление первым и вторым роботом через интерфейс оператора.

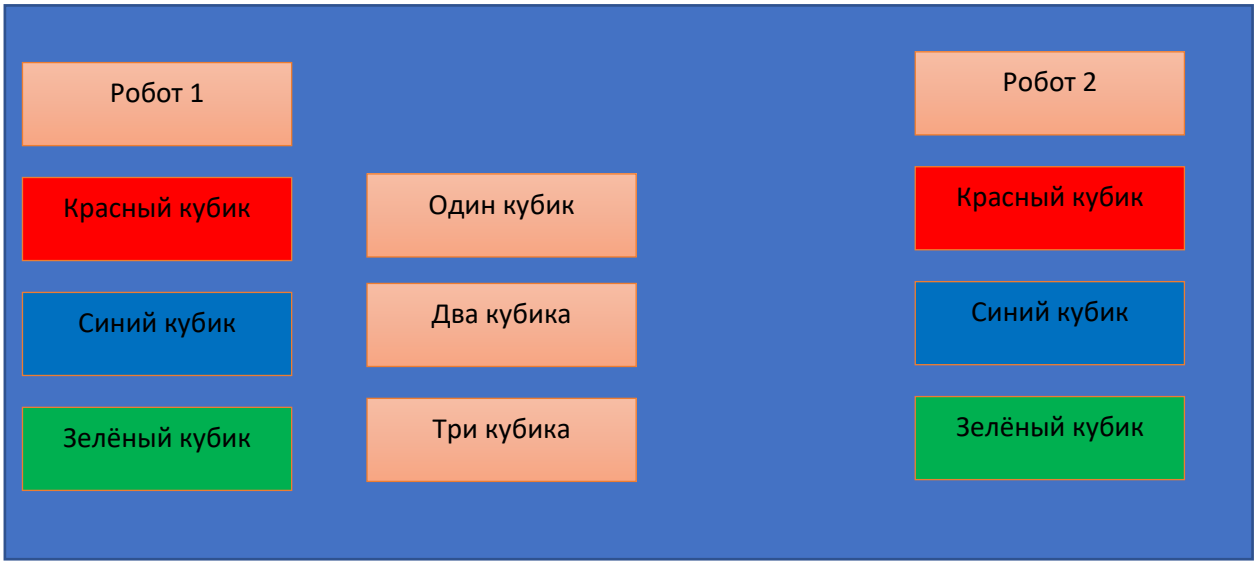
Красный – Кнопка. По нажатию робот перетаскивает красный кубик, а именно перемещает манипулятор, опускает его над красным кубиком, опускает, захватывает, поднимает, перемещает на нужную позицию, опускает манипулятор, отпускает кубик.

Зелёный – Кнопка. По нажатию робот перетаскивает красный кубик.

Синий – Кнопка. По нажатию робот перетаскивает синий кубик.

Parking – Кнопка. По нажатию соответствующий робот идёт на парковку и опускается вниз.

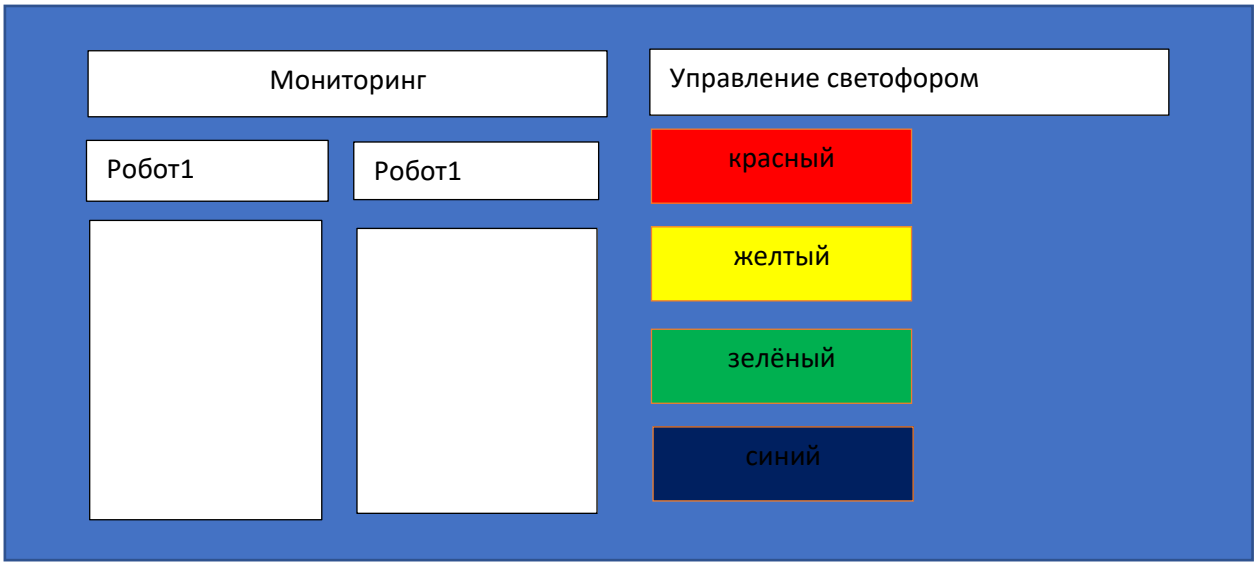
1 – Кнопка. По нажатию робот берёт один кубик со склада и переносит на место сбора.



2 – Кнопка. По нажатию робот берёт два кубика со склада и переносит на место сбора.

3 – Кнопка. По нажатию робот берёт три кубика со склада и переносит на место сбора.

Создайте интерфейс инженера. Организуйте просмотр информации о положении сервоприводов робота и др., а также ручное управление светофором.



Оборудование

В решении продемонстрирована работа с роботом и светофором, остальные компоненты набора не использовались. Основные ноды, которые применяются для управления роботом это IOT Function – позволяет поменять параметры робота, IOT Send – передаёт значения параметров роботу, Get IOT – позволяет принять информацию от робота. При использовании всех трёх нод надо указать свойство Robot_name для соединения с первым (Robot_1), вторым (Robot_2) роботами, сигнальными светофорами (TrafficLights_1) или другими устройствами.

Основные параметры IOT Function:

X,Y – координаты манипулятора в декартовой системе.

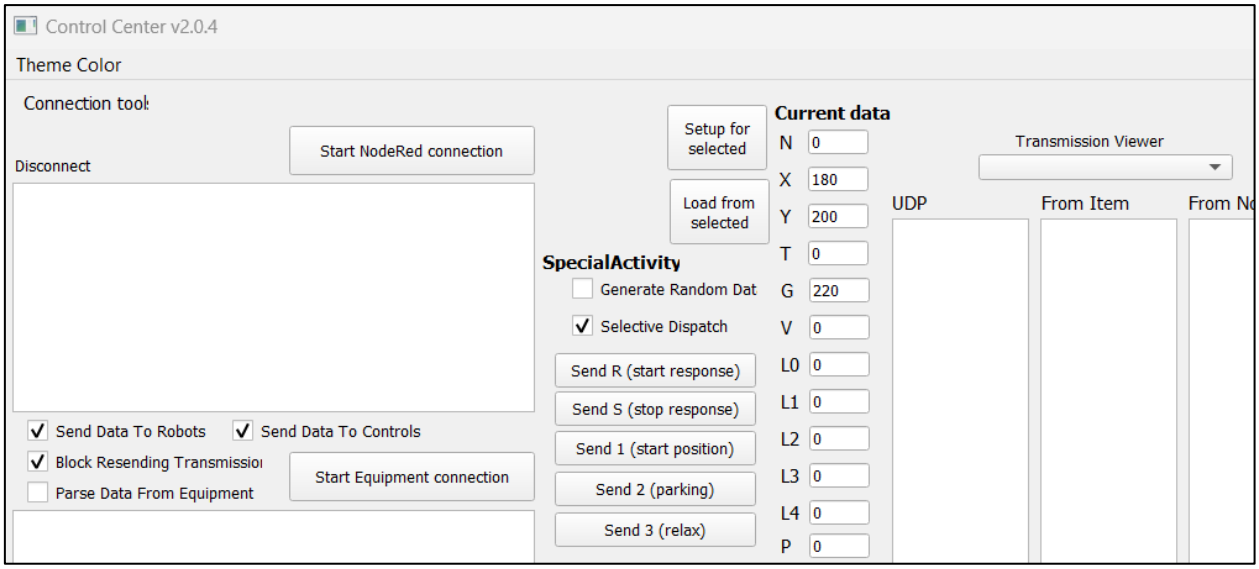
G – высота манипулятора.

V – захват (0 – без захвата, 1 – с захватом).

L1, L2, L3, L4 – признак включения цветов светофора (синий, красный, жёлтый, зелёный соответственно).

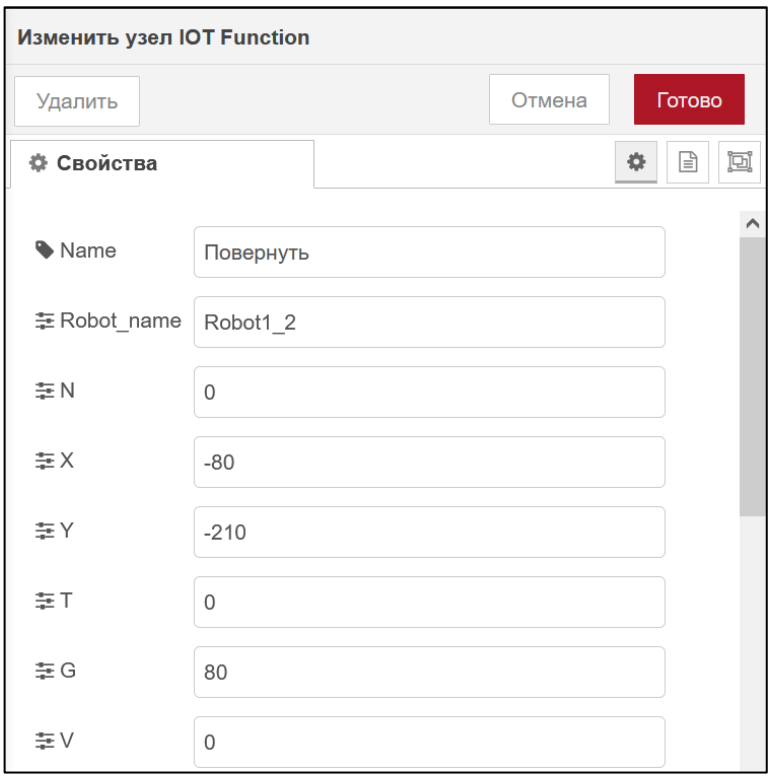
Буфером между роботом и программой служит ControlCenter (был использован IOT_ ControlCenter_v2.0.4). После запуска необходимо выбрать устройство (Transmission Vlewer) выполнить соединение с программой (Start

Node-red connection). Зелёные квадраты в Node-red и ControlCenter сигнализируют об успешной интеграции.



Меняя параметры узла IOT Function мы задаём устройству, название которого указано в свойстве Robot_name нужное изменение координат, высоты, захвата, цвета и т.д.

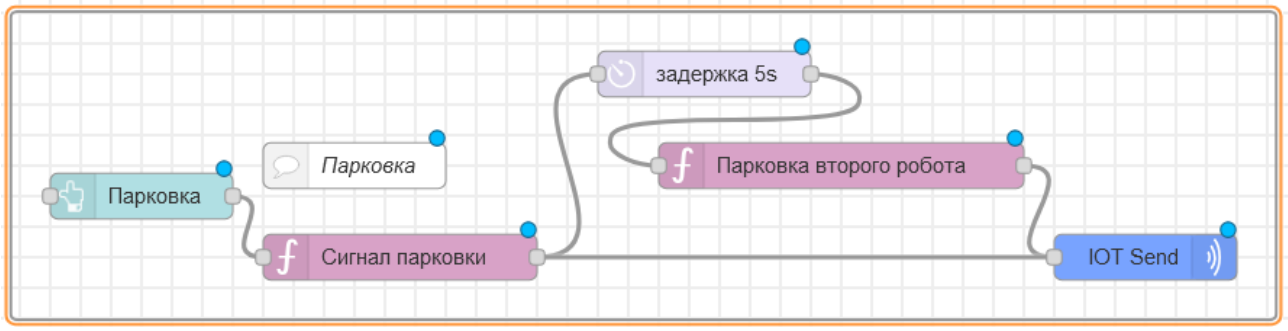
Понимая, что устройство мгновенно не сможет выполнить действие (поворот, захват и т.д.) необходимо задать время на исполнение. Это можно осуществить с помощью ноды delay, указав время задержки 3-5 секунд.

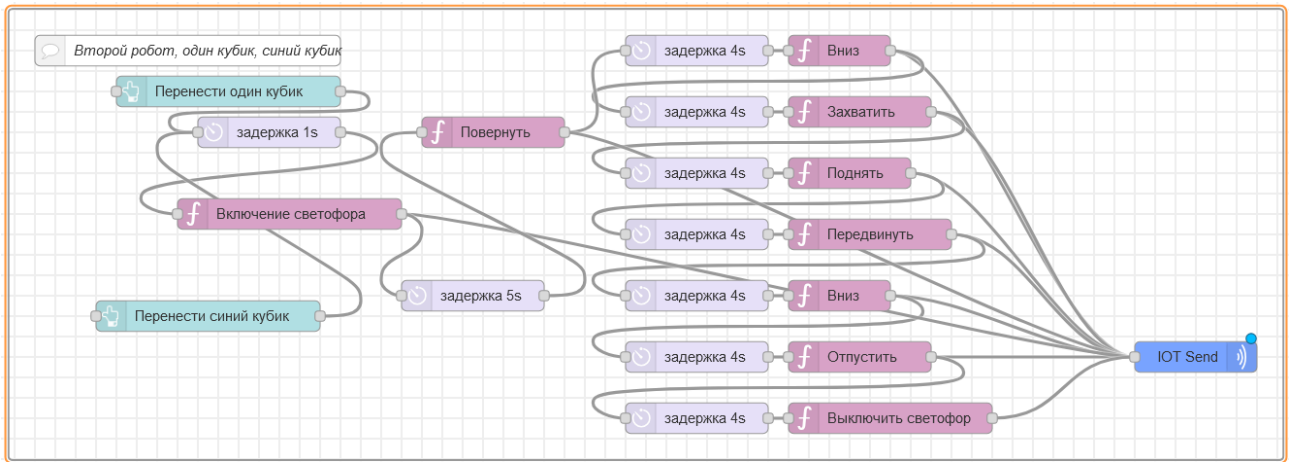


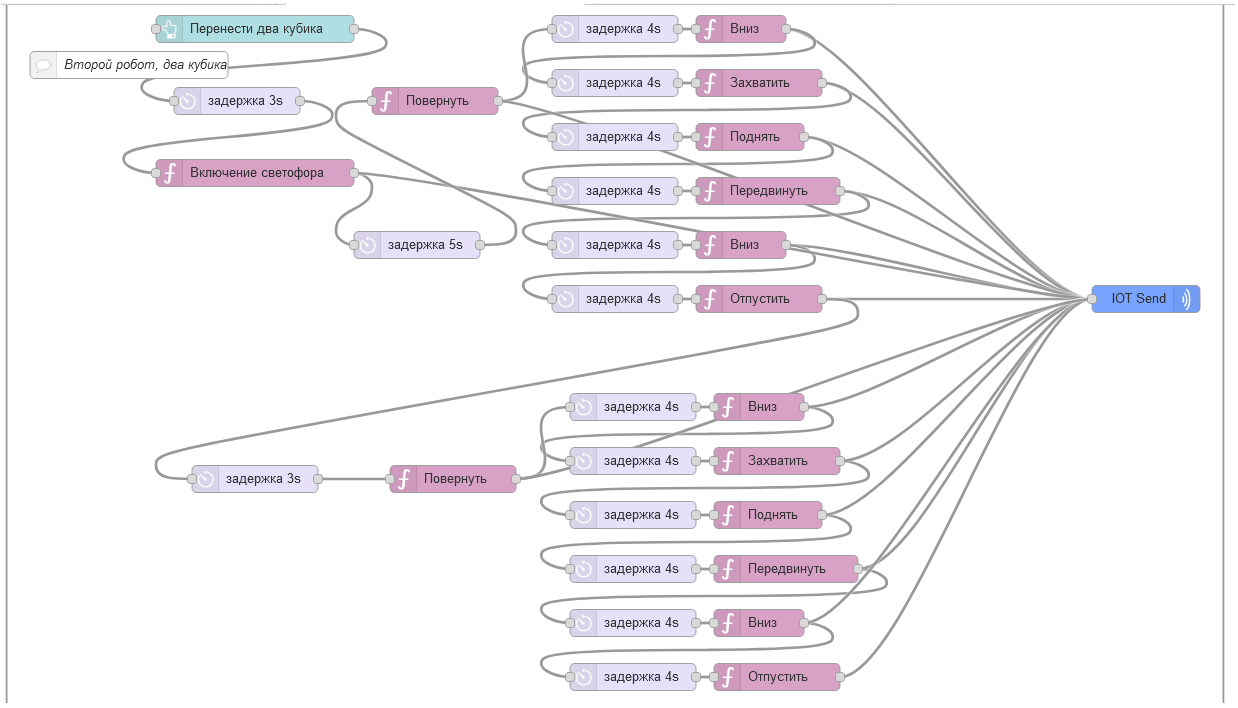
Например, чтобы переместить манипулятор, нужно изменить координаты X, Y. Чтобы опустить или поднять G, чтобы выполнить захват, параметру V нужно присвоить 1, а при отпускании 0. Необходимо следить за тем, чтобы не задеть соседние детали (контролируйте высоту).

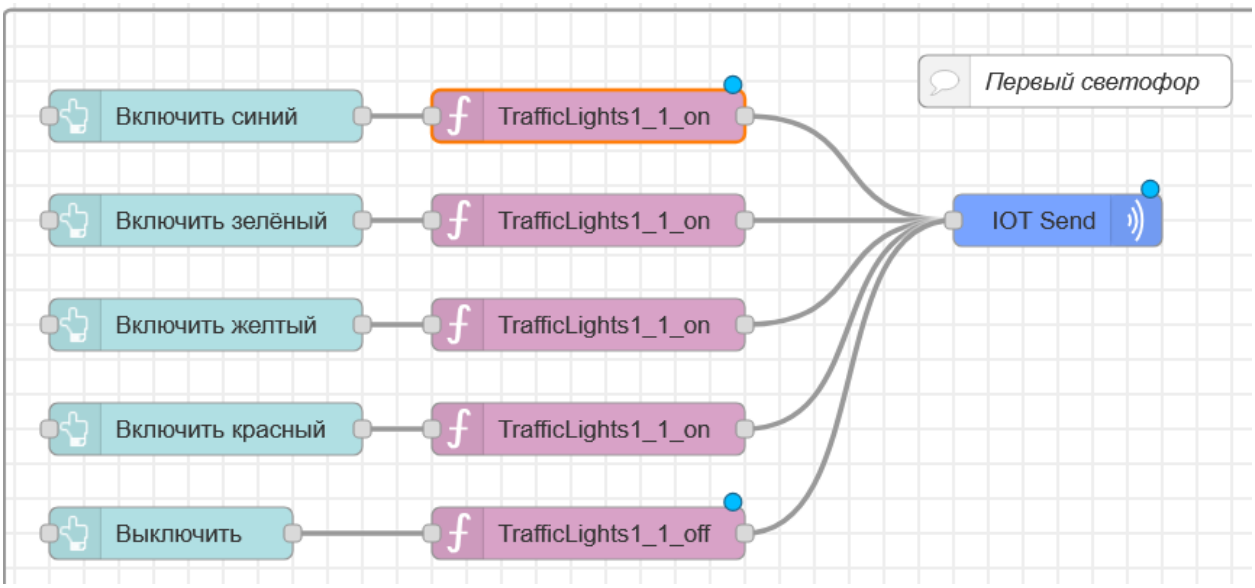
В представленных примерах чтобы опустить «руку» вниз нужно в параметрах функции 80 заменить на 20. Чтобы выполнить захват, все параметры оставить те же, кроме V, которому присвоить 1. Поднимаем манипулятор вверх (G), перемещаем на точку сборки (X, Y), опускаем (G), убираем захват (V) и т.д.

При управлении светофором нужно менять параметры L1, L2, L3, L4. Они могут принимать значения 1 (включено) или 0 (выключено).

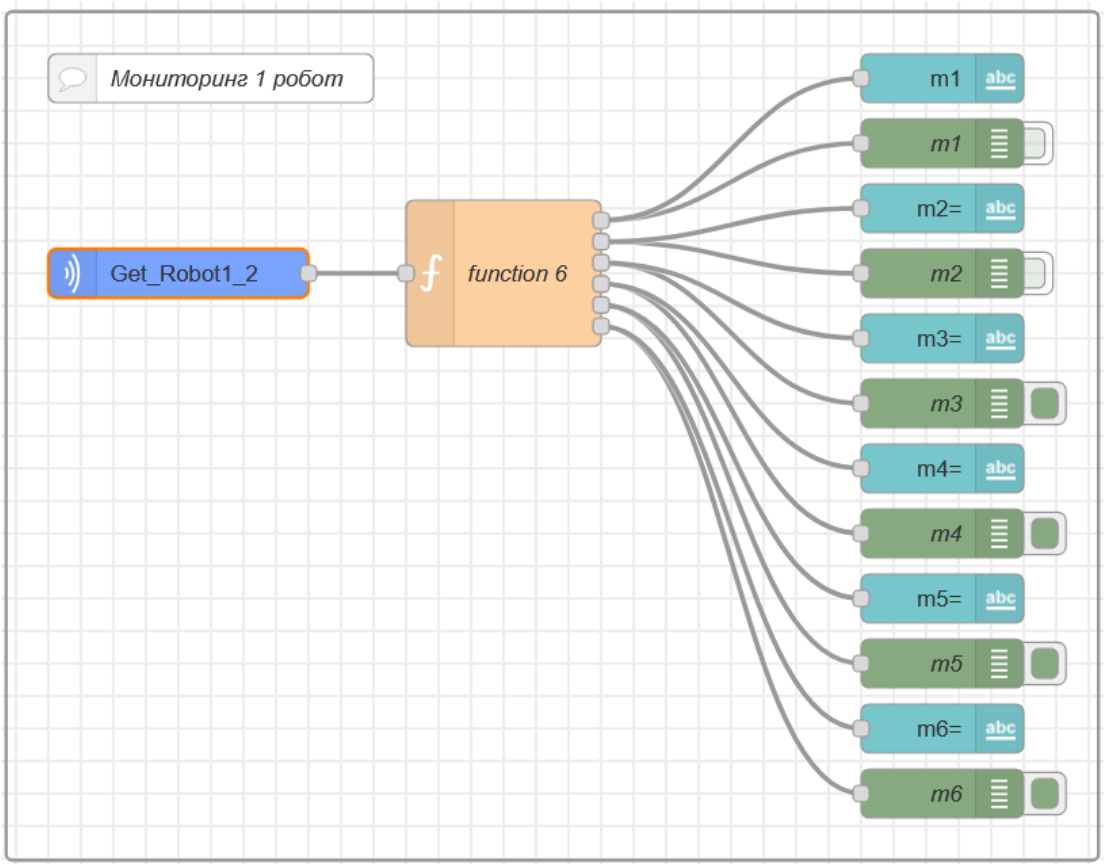


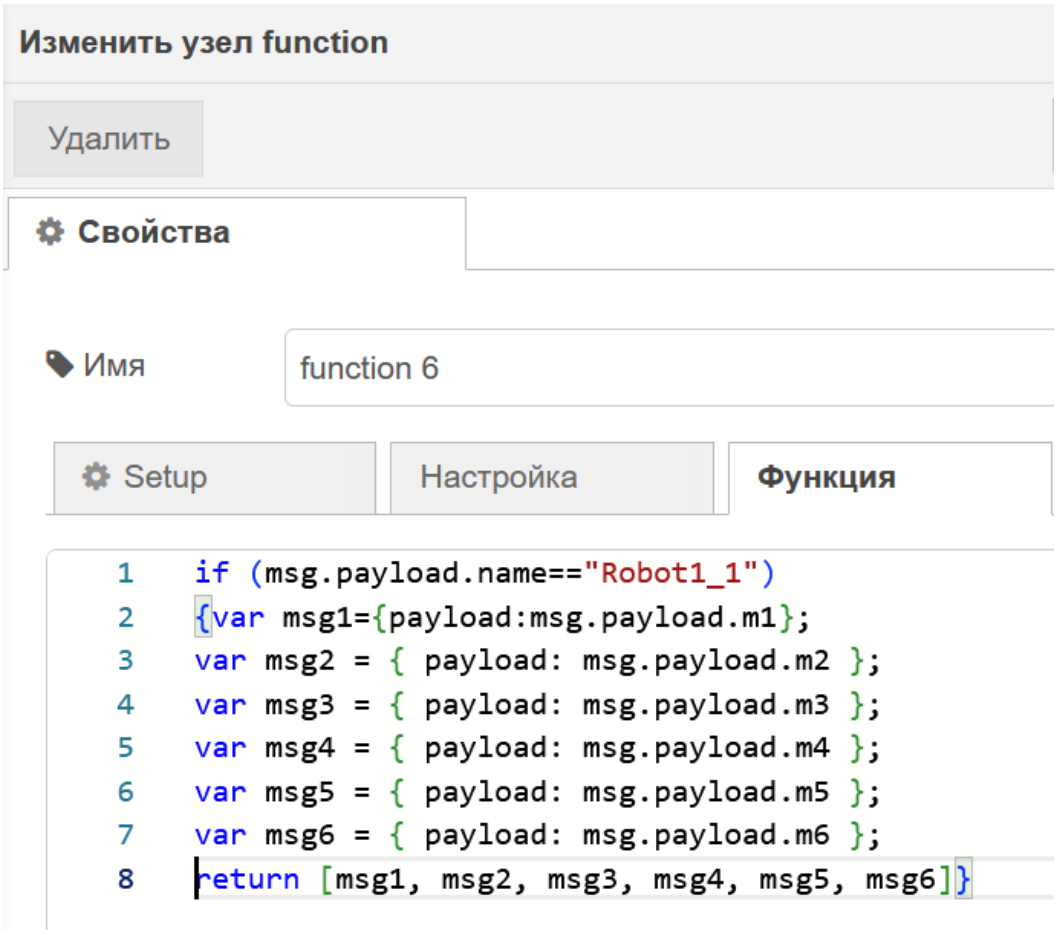




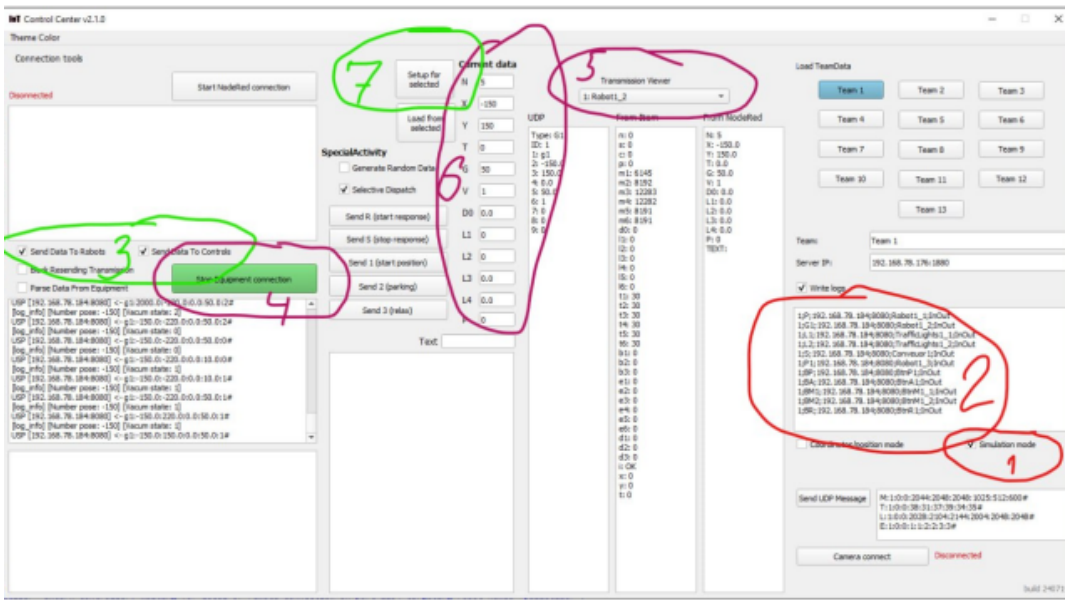


При мониторинге получаем информацию от робота и выводим её в разные текстовые поля. Дебаги оставили для тестирования.



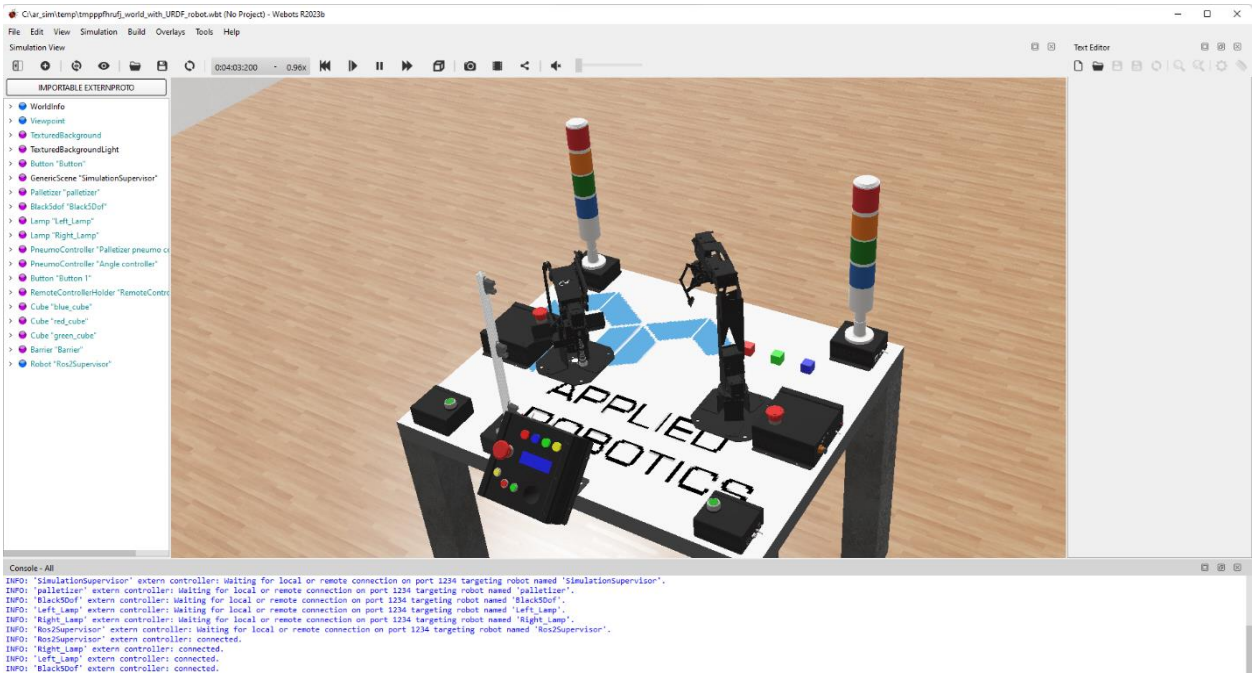


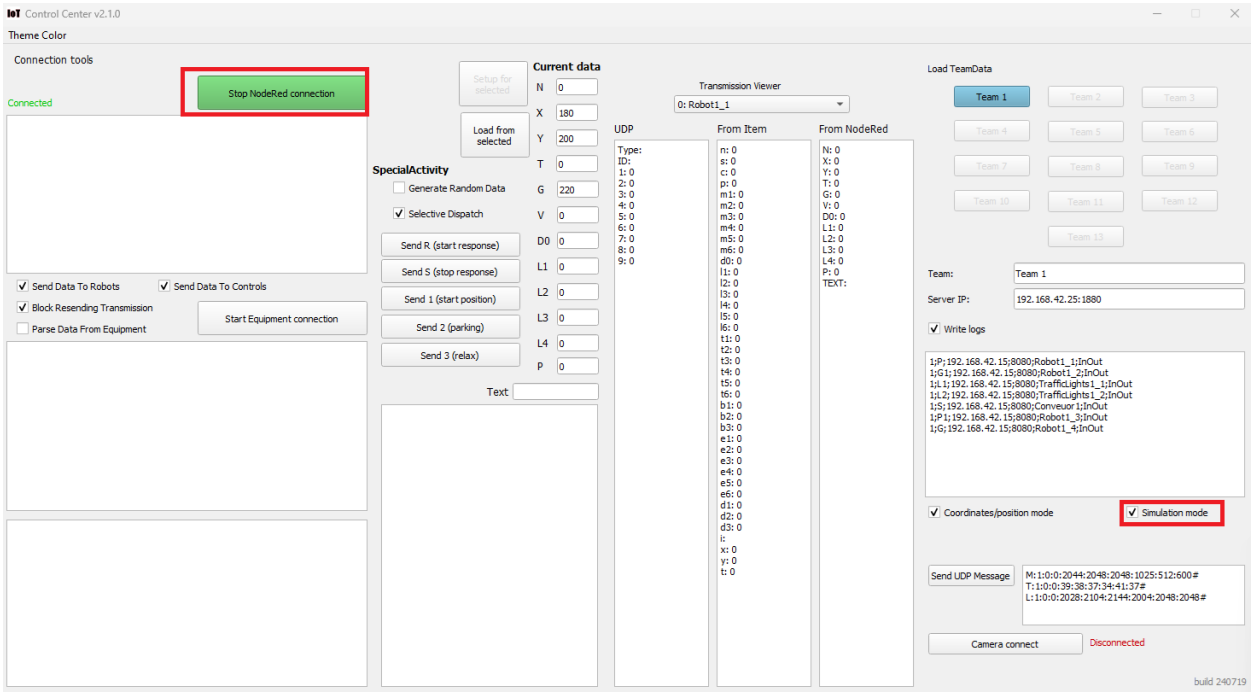
При подключении ControlCenter в режиме реального времени можно видеть изменения параметров через указанное время задержки.



1 - галочка для работы с симулятором
2 - отличаться должны только цифры 78.184, всё остальное должно быть также
3 - галочка для разрешения управления роботом вторая для разрешения управления смарт устройством
4 - нажимаем старт для управления устройствами
5 - выбираем устройство которым хотим управлять
6- задаем значения которые нужно отправить на выбранное устройство
важно помнить что в случае если значение вне диапазона разрешенных, то робот будет полностью игнорировать
7 - кнопка для отправки команд роботу 20:13

Существуют манипуляторы, созданные энтузиастами. Они позволяют заменить дорогостоящие устройства. Работая с манипуляторами, нет опасения сломать робота задав некорректные значения. Но, к сожалению, на данный момент в свободном распространении манипуляторов нет. Тот, который нам показывали в технопарке г. Калуги не распространяется, сложно настраивается, часто даёт сбои.


C:\ar_sim\temp\tmpppfhrufj_world_with_URDF_robot.wbt (No Project) - Webots R2023b
File Edit View Simulation Build Overlays Tools Help
Simulation View
Text Editor
IMPORTABLE EXTERNPROTO
WorldInfo
Viewpoint
TexturedBackground
TexturedBackgroundLight
Button "Button"
GenericScene "SimulationSupervisor"
Palletizer "palletizer"
Black5dof "Black5Dof"
Lamp "Left_Lamp"
Lamp "Right_Lamp"
Button "Button 1"
Cube "blue_cube"
Cube "red_cube"
Cube "green_cube"
Barrier "Barrier"
Robot "Ros2Supervisor"
APPLIED
ROBOTICS
Console - All
INFO: 'SimulationSupervisor' extern controller: Waiting for local or remote connection on port 1234 targeting robot named 'SimulationSupervisor'.
INFO: 'palletizer' extern controller: Waiting for local or remote connection on port 1234 targeting robot named 'palletizer'.
INFO: 'Black5Dof' extern controller: Waiting for local or remote connection on port 1234 targeting robot named 'Black5Dof'.
INFO: 'Left_Lamp' extern controller: Waiting for local or remote connection on port 1234 targeting robot named 'Left_Lamp'.
INFO: 'Right_Lamp' extern controller: Waiting for local or remote connection on port 1234 targeting robot named 'Right_Lamp'.
INFO: 'Ros2Supervisor' extern controller: Waiting for local or remote connection on port 1234 targeting robot named 'Ros2Supervisor'.
INFO: 'Ros2Supervisor' extern controller: connected.
INFO: 'Right_Lamp' extern controller: connected.
INFO: 'Left_Lamp' extern controller: connected.
INFO: 'Black5Dof' extern controller: connected.


IoT Control Center v2.1.0
Theme Color
Connection tools
Connected
Stop NodeRed connection
Send Data To Robots
Send Data To Controls
Block Resending Transmission
Parse Data From Equipment
Start Equipment connection
Setup for selected
Load from selected
SpecialActivity
Generate Random Data
Selective Dispatch
Send R (start response)
Send S (stop response)
Send 1 (start position)
Send 2 (parking)
Send 3 (relax)
Text
Current data
N 0
X 180
Y 200
T 0
G 220
V 0
D0 0
L1 0
L2 0
L3 0
L4 0
P 0
Transmission Viewer
0: Robot1_1
UDP
Type:
ID:
1: 0
2: 0
3: 0
4: 0
5: 0
6: 0
7: 0
8: 0
9: 0
From Item
n: 0
s: 0
c: 0
p: 0
m1: 0
m2: 0
m3: 0
m4: 0
m5: 0
m6: 0
d0: 0
l1: 0
l2: 0
l3: 0
l4: 0
l5: 0
l6: 0
t1: 0
t2: 0
t3: 0
t4: 0
t5: 0
t6: 0
b1: 0
b2: 0
b3: 0
e1: 0
e2: 0
e3: 0
e4: 0
e5: 0
e6: 0
d1: 0
d2: 0
d3: 0
i:
x: 0
y: 0
t: 0
From NodeRed
N: 0
X: 0
Y: 0
T: 0
G: 0
V: 0
D0: 0
L1: 0
L2: 0
L3: 0
L4: 0
P: 0
TEXT:
Load TeamData
Team 1
Team 2
Team 3
Team 4
Team 5
Team 6
Team 7
Team 8
Team 9
Team 10
Team 11
Team 12
Team 13
Team:
Team 1
Server IP:
192.168.42.25:1880
Write logs
1;P;192.168.42.15;8080;Robot1_1;InOut
1;G1;192.168.42.15;8080;Robot1_2;InOut
1;L1;192.168.42.15;8080;TrafficLights1_1;InOut
1;L2;192.168.42.15;8080;TrafficLights1_2;InOut
1;S;192.168.42.15;8080;Conveuor1;InOut
1;P1;192.168.42.15;8080;Robot1_3;InOut
1;G;192.168.42.15;8080;Robot1_4;InOut
Coordinates/position mode
Simulation mode
Send UDP Message
M:1:0:0:2044:2048:2048:1025:512:600#
T:1:0:0:39:38:37:34:41:37#
L:1:0:0:2028:2104:2144:2004:2048:2048#
Camera connect
Disconnected
build 240719

Практическая работа 5. Подключение и работа с базами данных

Цель: реализовать подключение и работу Node-red в базе данных mysql.

План работы:

В этом разделе вы узнаете, как с помощью среды Node-RED работать с базой данных:

– загрузить и настроить ноду MySql;

– создать базу данных;

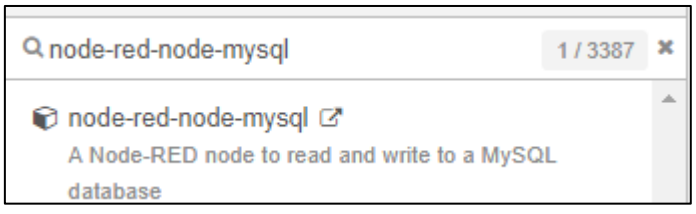
– создать таблицу данных, организовать выборку;

– добавить информацию в.

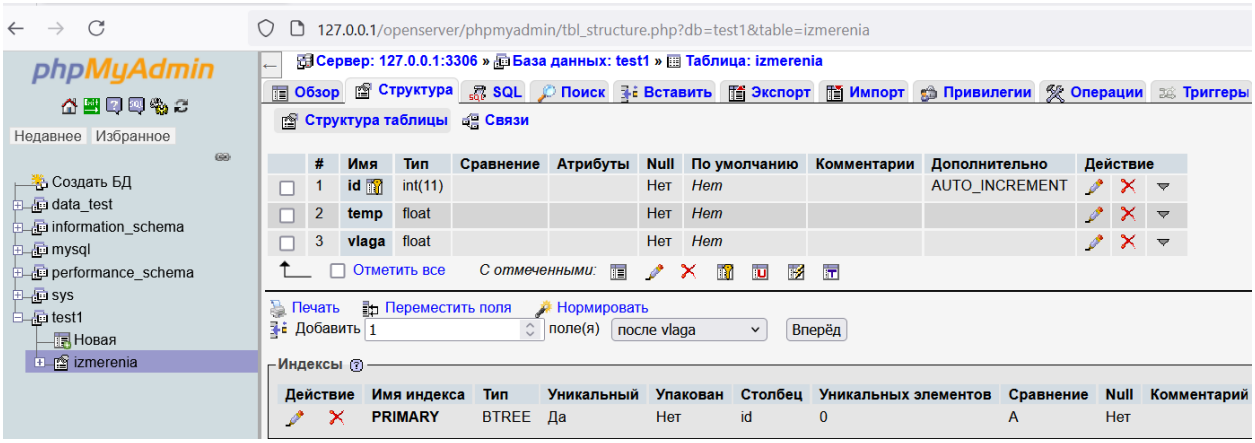
Хо

Node-red реализует подключение к базе данных mysql. Чтобы реализовать подключение к базе данных Node-Red (MySQL), сначала вам нужно загрузить программное обеспечение MySQL на свой компьютер и установить Navicat для удобного просмотра баз данных. Также необходимо знать язык запросов SQL.

Добавьте библиотеку, позволяющую работать с базами данных через Управление палитрой.

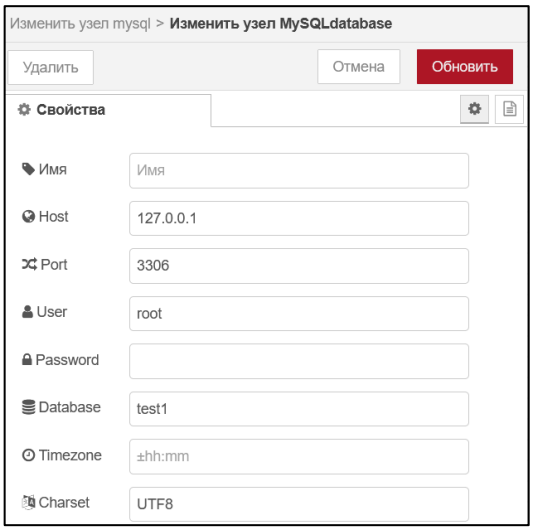


Для создания базы данных можно воспользоваться разными сервисами. Например PhpMyAdmin, который входит в OpenServer, или MySqlWorkBench.



Заполните таблицу несколькими строками

Настройте ноду mysql на созданную базу данных. После сохранения проверьте, что соединение установлено. Это видно по зелёному квадрату возле ноды mysql.

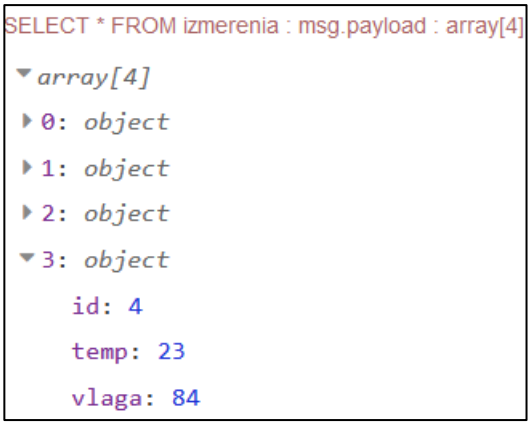


Поиск в таблице данных

Потребуются ноды **inject, function, mysql, debug.**

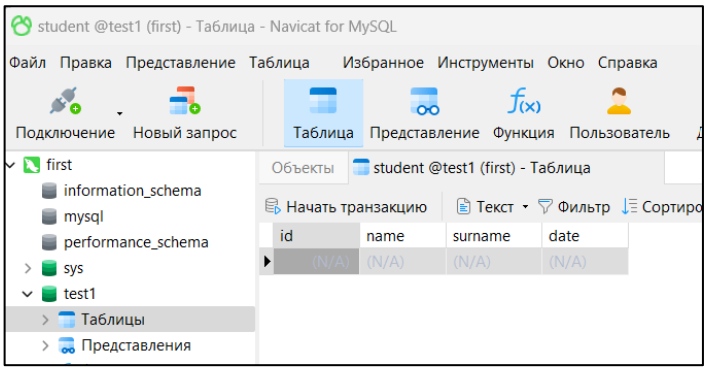

```
msg.topic = "SELECT * FROM izmerenia";
return msg;
```

Щёлкните по квадрату на ноде **inject**. Проверьте через отладчик, что все объекты найдены.



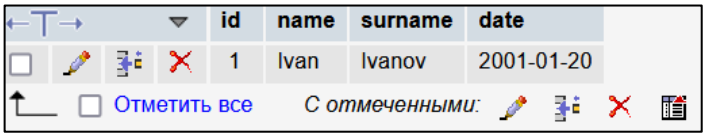
Создание таблицы Student

```
msg.topic = "CREATE TABLE IF NOT EXISTS `student`(  `id` INT UNSIGNED AUTO_INCREMENT,  `name` VARCHAR(100) NOT NULL,  `surname` VARCHAR(40) NOT NULL,  `date` DATE,  PRIMARY KEY ( `id` ))ENGINE=InnoDB DEFAULT CHARSET=utf8";
return msg;
```



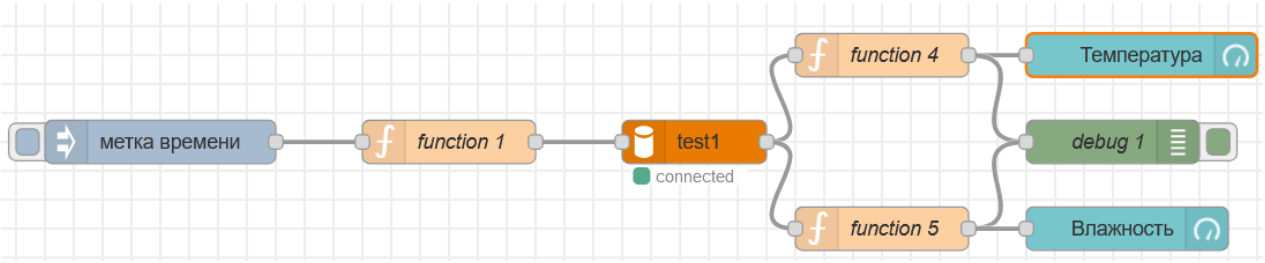
Добавление информации о студенте

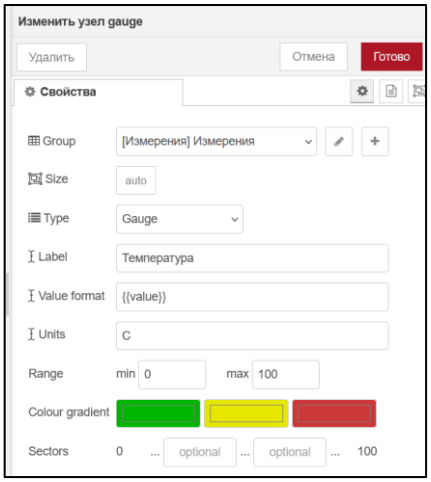
```
msg.topic = "INSERT INTO student(name, surname, date) VALUES ('Ivan', 'Ivanov', '1.01.2000')";
return msg;
```



Создание dashboard для отображения информации с базы данных

Настройка ноды gauge



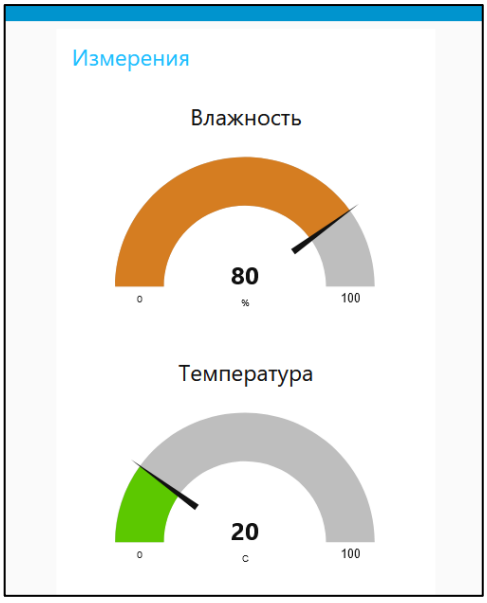


Функции

```
msg.payload=msg.payload[0].temp;
return msg;
```

```
msg.payload=msg.payload[0].vlaga;
return msg;
```

Результат



Практическая работа 6. Node-red и Arduino

Node-RED - это инструмент для объединения аппаратных устройств, API и онлайн-сервисов новыми и интересными способами.

Цель практической работы: объединить Node-red и Arduino. Научиться передавать данные от контроллера в DashBoard и управлять из Node-red элементом (светодиодом).

Ход работы

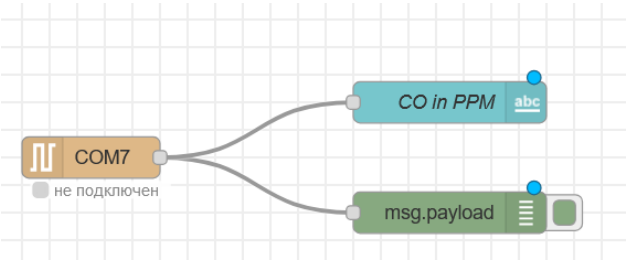
Подготовка платы Arduino

1. Подключите любой аналоговый датчик к порту A0 (я работала с фоторезистором).
2. Скорость передачи данных в бодах должна быть 57600.
3. Не используйте / не открывайте последовательный монитор, держите COM-порт свободным.
4. Подключите плату arduino к компьютеру.
5. Напишите код считывания показаний датчика.

```
const int AOUTpin= 0;//the AOUT pin of the CO sensor goes into analog pin A0 of the arduino
int value;
void setup()
{
Serial.begin(57600);//sets the baud rate
pinMode(AOUTpin, INPUT);//sets the pin as an input to the arduino
}
void loop()
{
```

```
value= analogRead(AOUTpin);//reads the analaog value from the CO sensor's AOUT pin
Serial.print("CO value: ");
Serial.println(value);//prints the CO value
delay(60000);
}
```

Подключите палитру node-red-node-serialport



Настройте узел с помощью вашего COM-порта, к которому подключен Arduino. Это делается вручную, в моем случае это COM7.

Для текстовой ноды создайте dashboard. Отобразите на нём показания фоторезистора. Можно дополнить проект, записав показания в таблицу базы данных.

Есть и другие способы настроить взаимодействие между Arduino и Node-RED.

Последовательная коммуникация

Поскольку платы Arduino – это устройства, «общающееся» при помощи последовательной коммуникации, для коммуникации с ними можно использовать ноды для последовательной передачи (туда и обратно) данных.

Последовательная коммуникация – очень распространенный способ программирования Arduino с помощью IDE, т.к. позволяет отправлять и получать данные по последовательному порту для взаимодействия с созданным вами проектом. Убедитесь лишь, что у вас выставлена одинаковая скорость коммуникации на обоих концах последовательного порта.

Обратите внимание! Вы не сможете одновременно использовать IDE Arduino и Arduino, т.к. они будут конфликтовать друг с другом. Таким образом, если вам нужно будет перепрограммировать Arduino при помощи IDE Arduino, то вам сначала нужно будет остановить Node-RED.

Firmata

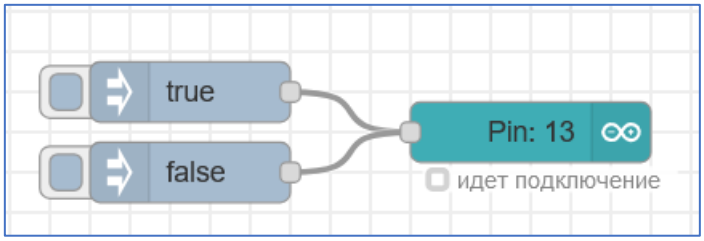
Firmata – это протокол для коммуникации между Arduino (или другим микроконтроллером) и компьютером-хостом, обеспечивающий прямой доступ к входным/выходным контактам.

Установка

Сначала вам нужно будет загрузить на Arduino стандартный скетч Firmata при помощи стандартных инструментов загрузки, имеющихся в IDE Arduino. Как правило, для этого в IDE Arduino нужно просто кликнуть на «Файл» > «Примеры» > «Firmata» > «StandardFirmata» (File > Examples > Firmata > StandardFirmata).

Затем вам нужно будет установить в палитру (Palette) Node-RED ноды для Arduino node-red-node-arduino.

Создайте поток, который будет включать и выключать светодиод, подключённый к 13 пину (этот светодиод интегрирован на плату).



Этот поток настроен на автоматическое определение платы, подключенной к последовательному порту. Если вам нужно поменять порт, сделайте двойной клик по Arduino-ноде «Pin 13», затем кликните на кнопку с иконкой карандаша и задайте нужный порт.

Затем кликните на кнопку «Deploy», после чего поток должен начать работать. Светодиод на 13 контакте должен начать включаться и выключаться по нажатию соответствующей ноды.

Что умеют делать Arduino-ноды

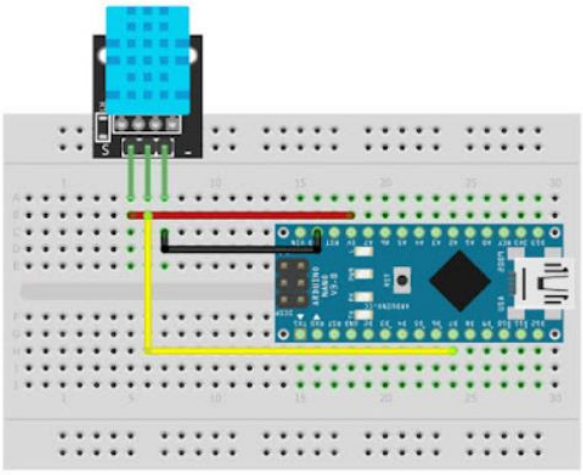
Выходная нода Arduino в данный момент поддерживает три режима работы:

- Цифровые данные («0» или «1»)
- Аналоговые данные (от «0» до «255»)
- Сервопривод (от «0» до «180»)

Входная нода Arduino (она не используется в потоке выше) поддерживает работу и с цифровыми, и аналоговыми контактами. Она отправляет сообщение всякий раз, когда обнаруживает какое-либо изменение состояния контакта. Это не страшно, если вы работаете с цифровыми контактами, т.к. они, как правило, достаточно стабильны, но данные на аналоговых контактах склонны меняться с большой частотой (по умолчанию – 40 раз в секунду...). Но в настройках последовательного порта ее можно понизить до более приемлемого уровня.

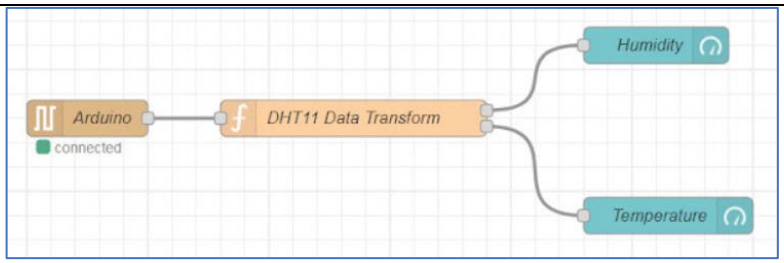
Задания для самостоятельной работы

Задание 1. Отобразите в Dashboard показания температуры и влажности с помощью ноды Gauge.



```
#include <DHT.h>        // Include Adafruit DHT11 Sensors Library

#define DHTPIN 7        // DHT11 Output Pin connection
#define DHTTYPE DHT11   // DHT Type is DHT11
```

```
DHT dht(DHTPIN, DHTTYPE);   // Initialize DHT sensor

void setup () {
  dht.begin();
  Serial.begin(9600);         // To see data on serial monitor
}

void loop (){

    float H = dht.readHumidity();     //Read Humidity
    float T = dht.readTemperature();    // Read temperature as Celsius

    // Check if any reads failed and if exit
    if (isnan(H) || isnan(T)){
      Serial.println("Failed to read from DHT sensor!");
      return;
    }

    // Combine Humidity and Temperature into single string
    String dhtData = String(H) + "," + String(T);
    Serial.println(dhtData);
    delay(2000);   // Wait two seconds between measurements
}
```

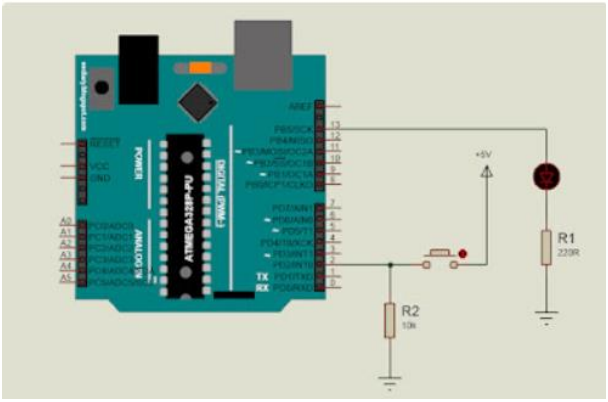



```
m = msg.payload.split(',');
H = {payload:parseFloat(m[0])};
T = {payload:parseFloat(m[1])};
```

return [H,T];

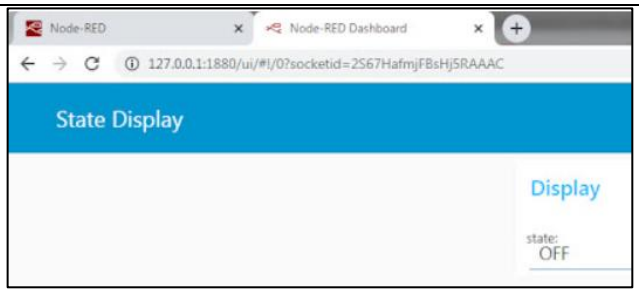
Организуйте запись показаний в базу данных.

Задание 2. Отобразите в Dashbord информацию о состоянии светодиода.




```
const int buttonPin = 2;     // the number of the pushbutton pin
const int ledPin =  13;      // the number of the LED pin

// variables will change:
int buttonState = 0;         // variable for reading the pushbutton status

void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
  Serial.begin(9600);
```

```
}

void loop() {
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed. If it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
    Serial.println("ON");
  } else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
    Serial.println("OFF");
  }
}
```



Список использованных источников

1. Внутренний инженер Node-RED: урок по базовым нодам | Умный дом (видео). https://rutube.ru/video/fffeec1b19e571d47d70d63deb6d5d11/
2. Конспекты занятий, записи, примеры из лекций, записанных на занятиях в рамках стажировки в Федеральном центре профессионального образования г. Калуга, компетенция «Интернет вещей», преподаватель Байшегурова Розалия Рифовна.
3. Сайт Node-red.org.
4. Создание панелей (Dashboards) в Node-RED
   https://wirenboard.com/wiki/index.php?title=Node-RED_Dashboards&mobileaction=toggle_view_desktop
5. GokulP3 (India) Data Logging With Node-RED and Arduino
   https://www.instructables.com/Data-Logging-With-Node-RED-and-Arduino/
6. Joseph Chege
   How to Connect Node-Red, Save|Select|Insert Data to MySQL Example Tutorial
7. Learn how to use Node-RED together with the Arduino Cloud.
   Author Liam Aljundi, https://docs.arduino.cc/arduino-cloud/guides/node-red/
8. Node-RED:Введение/Взаимодействие с Arduino
   Перевод: Максим Кузьмин
   Проверка/Оформление/Редактирование: Мякишев Е.А.
   https://wikihandbk.com/wiki/Node-RED:%D0%92%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5/%D0%92%D0%B7%D0%B0%D0%B8%D0%BC%D0%BE%D0%B4%D0%B5%D0%B9%D1%81%D1%82%D0%B2%D0%B8%D0%B5_%D1%81_Arduino