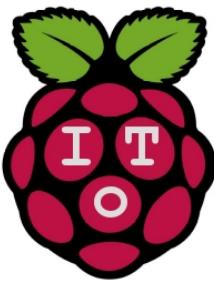


Internet of Things



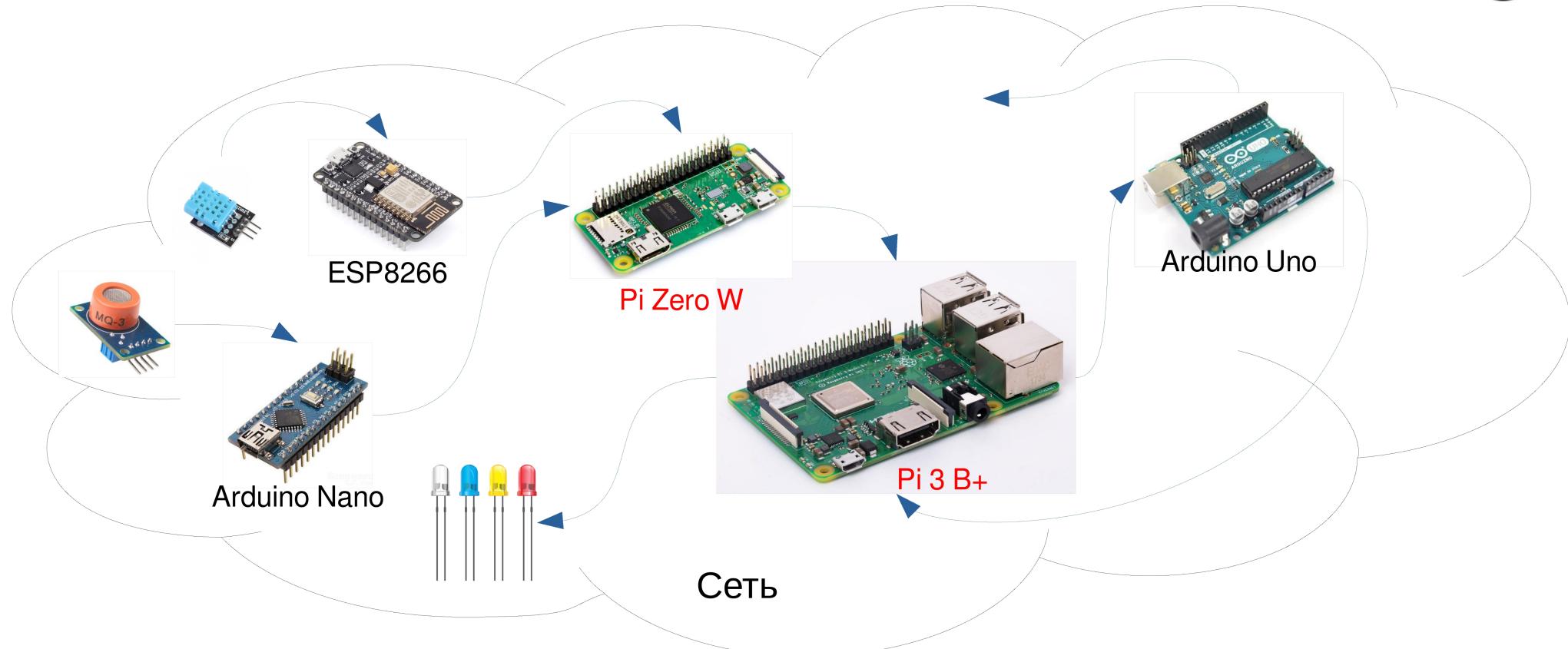
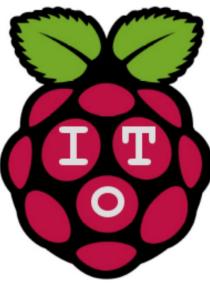
MQTT в сетях IoT

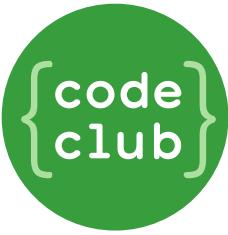
Шадринск
2018-2019

M. B. Шохирев

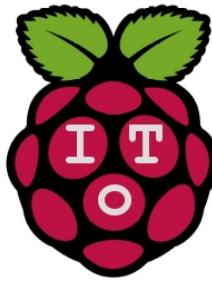
{code
club}

ИОТ = сеть контроллеров!





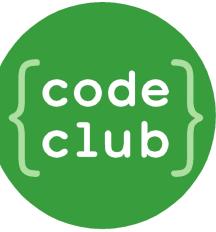
Сети IoT: всё плохо!



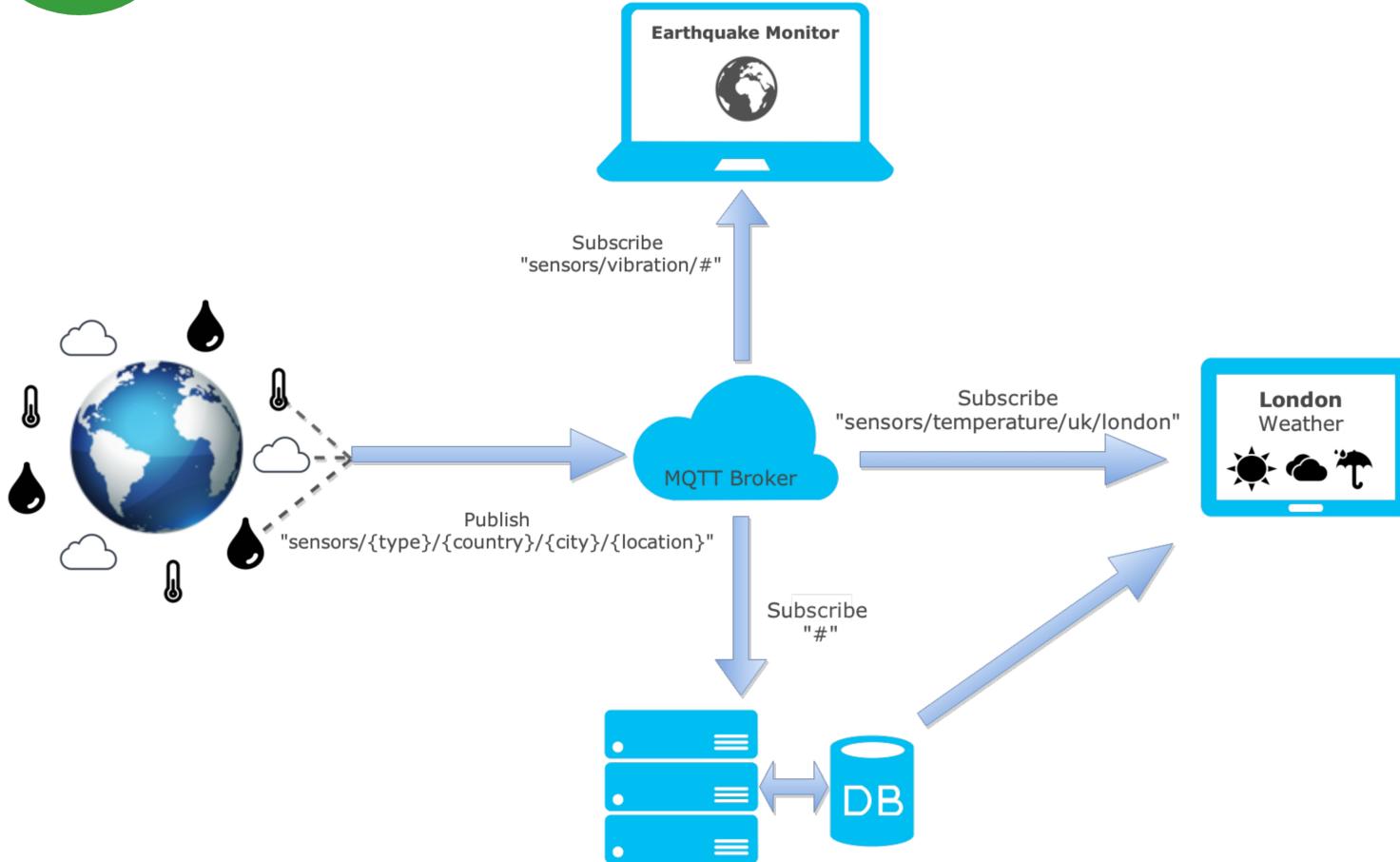
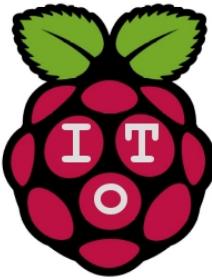
В сетях IoT типичны такие трудности:

- используются различные аппаратные платформы
- используются различные программные средства
- необходимо увязывать решения от разных производителей
- применяются разные сетевые технологии
- сеть состоит из отдельных удалённых фрагментов
- связь нестабильна: пропадает радио-сигнал и т. п.
- некоторые узлы сети могут быть временно недоступны
- случаются перебои в электроснабжении оборудования
- необходимо экономить заряд аккумулятора

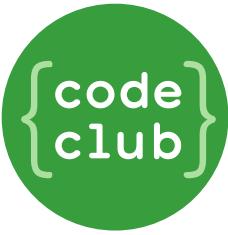
Требуется повышенная надёжность работы: исправление ошибок, повторная отправка данных, дублирование каналов передачи данных, резервирование узлов сети, самовосстановляемость систем, защита от злоумышленников.



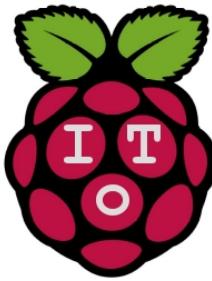
MQTT



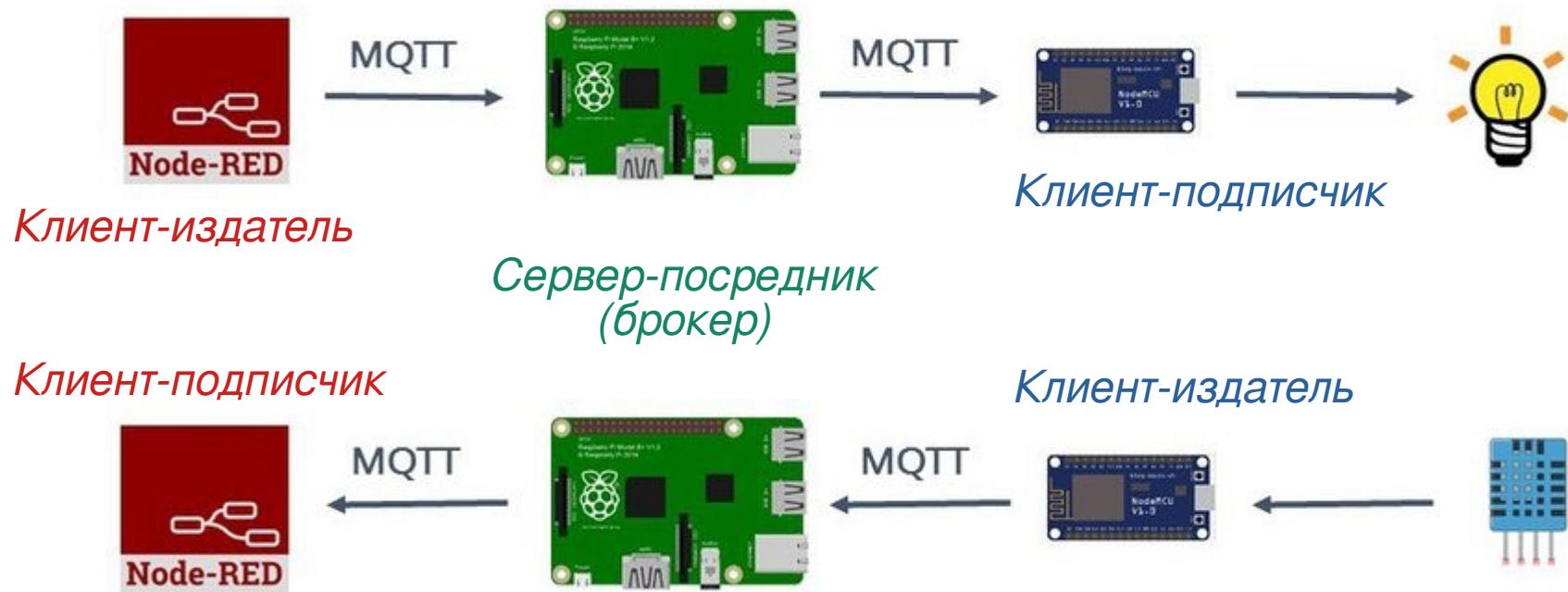
MQTT (Message Queuing Telemetry Transport) — сетевой протокол для передачи телеметрии при межмашинном взаимодействии (M2M = Machine-to-Machine). Его легко освоить и просто применять, он не сильно загружает каналы передачи данных, надёжен в работе при частых потерях связи, легко встраивается в любую систему, кроссплатформенный.

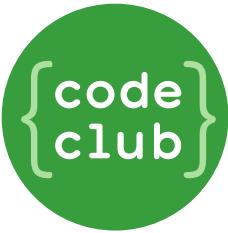


Модель MQTT

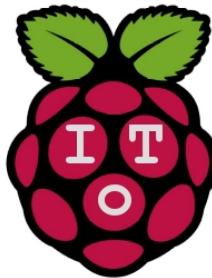


Протокол **MQTT** (Message Queuing Telemetry Transport) реализует обмен сообщениями по принципу «издатель-подписчик» с использованием посредника-брюкера.



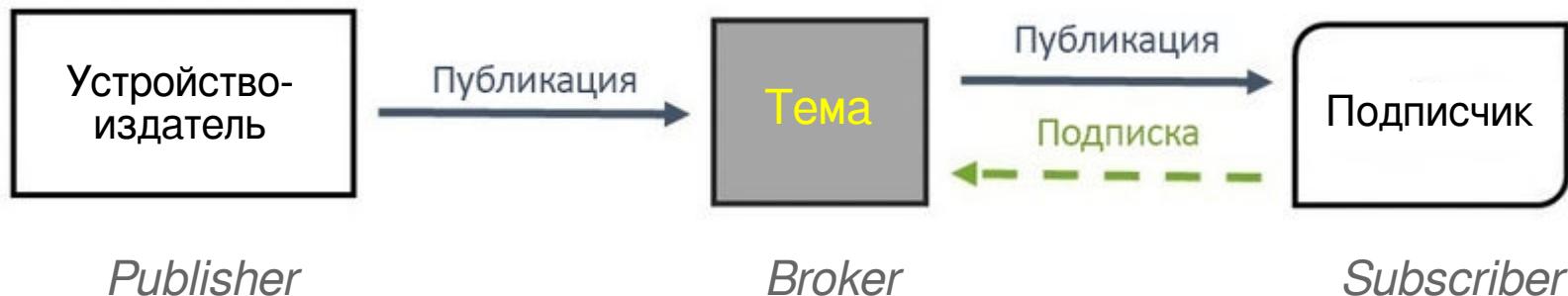


Издатель-подписчик

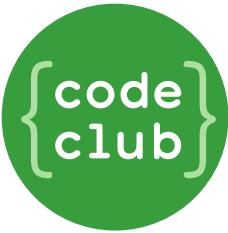


Модель «издатель-подписчик» означает:

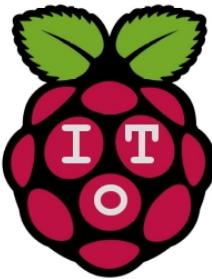
- одно устройство может публиковать (publish) сообщения для других устройств;
- устройство может подписаться (subscribe) на какую-нибудь тему (topic), чтобы получать интересующие его сообщения.



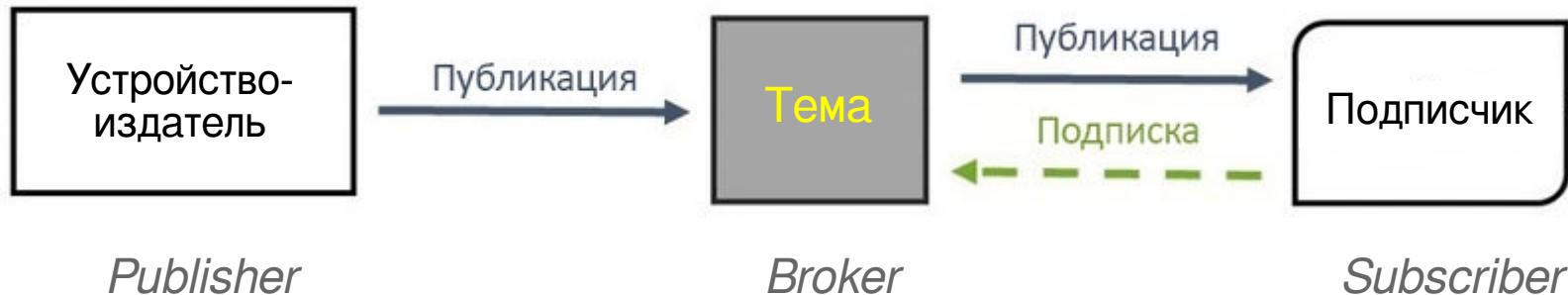
Когда одно устройство-издатель публикует сообщение в теме, а второе устройство-подписчик подписано на эту тему, то сообщение, опубликованное первым устройством, будет получено вторым устройством.



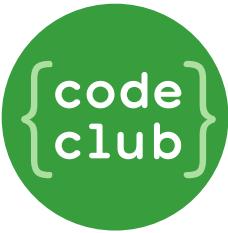
Темы



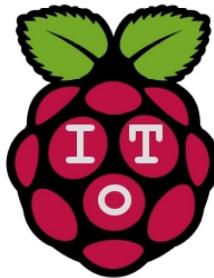
Темы (topic) — это способ группировать и структурировать сообщения, передаваемые через брокер. Имена тем и подтем разделяются символом «/». При именовании тем учитывается регистр букв (заглавные и строчные).



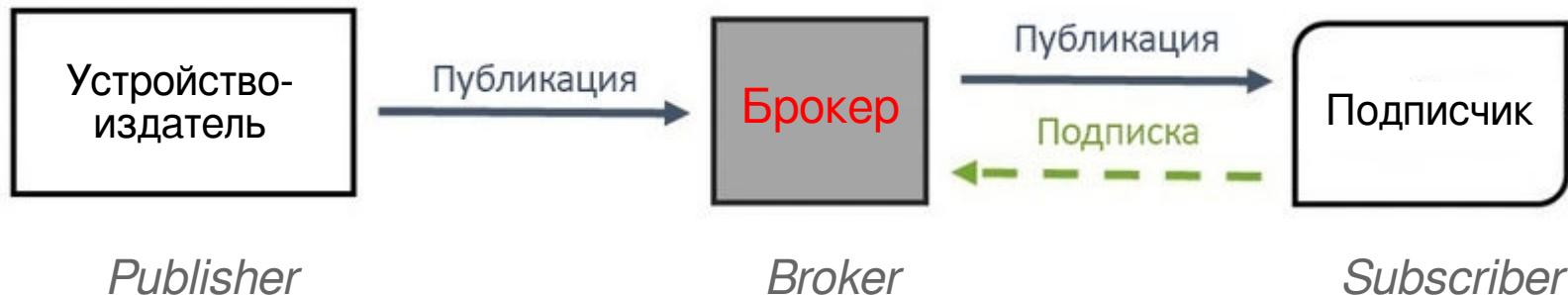
Темы могут называться строками на национальных языках, например, на русском, в кодировке UTF-8. Но рекомендуется называть темы осмысленными словами на английском языке в латинице.



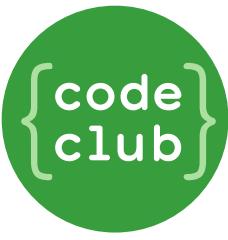
Брокер (посредник)



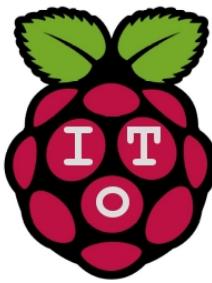
Брокер (Broker) – программа, которая организует публикацию сообщений в темах, получение сообщений от издателей, их отбор по подпискам и отправку подписчикам.



Имеется много разных программ-брокеров MQTT. Один из наиболее популярных брокеров, Mosquitto, реализован для Raspberry Pi.



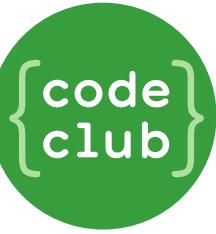
Управление по MQTT



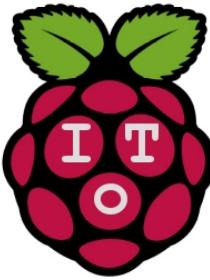
Например, чтобы включить светильник в офисе, нужно в управляющей клиентской программе, например, Node-RED на Raspberry Pi, опубликовать сообщение «ON» в теме `home/office/lamp` на брокере (который также развернут на Raspberry Pi).

Исполняющее клиентское устройство, например, модуль ESP8266, должно быть подписано на эту тему, чтобы получать сообщения, и когда оно получит опубликованное сообщение «ON», оно включит светильник.





Mosquitto

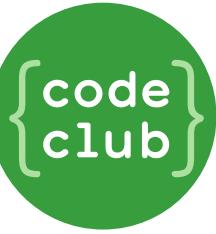


Mosquitto™ — брокер MQTT с открытым исходным кодом, развиваемый организацией Eclipse Foundation. Это легковесный сервер, который обеспечивает обмен сообщениями по протоколу MQTT. Он прекрасно подходит для обмена данными в сетях IoT между датчиками с низким энергопотреблением и мобильными устройствами, встраиваемыми компьютерами или микроконтроллерами. Он реализован для многих платформ, включая Raspberry Pi.

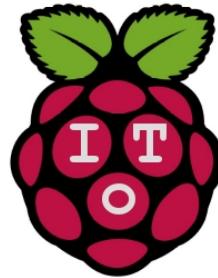
Проект **Mosquitto** также предоставляет библиотеку на языке C для реализации клиентов MQTT, а также популярные команды `mosquitto_pub` и `mosquitto_sub` для работы из командной строки.

0. Установить сервер и клиентов Mosquitto:

```
sudo apt-get update  
sudo apt-get install mosquitto mosquitto-clients
```



Mosquitto ← команды



A screenshot of a terminal window titled "pi@RPi06: ~". It shows two separate sessions. The top session is a subscription to the topic "sensor/temperature" with the command:

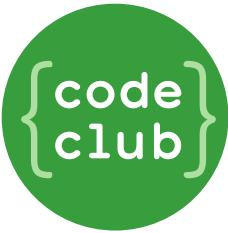
```
mosquitto_sub -h localhost -t "sensor/temperature"
```

The bottom session shows two publications to the same topic with different messages:

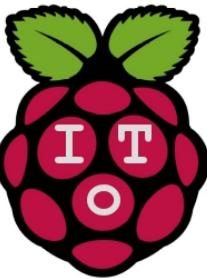
```
mosquitto_pub -h localhost -t "sensor/temperature" -m 23.5  
mosquitto_pub -h localhost -t "sensor/temperature" -m 23.4
```

1. Подписаться (SUBscribe) на нужную тему:
`mosquitto_sub -h localhost -t "sensor/temperature"`

2. Опубликовать (PUBLISH) в нужной теме:
`mosquitto_pub -h localhost -t "sensor/temperature" -m 23.5`
`mosquitto_pub -h localhost -t "sensor/temperature" -m 23.4`



Mosquitto ← bash



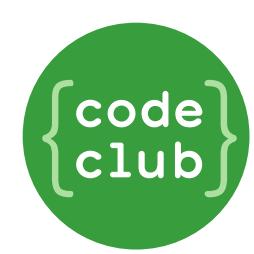
Подписаться (SUBscribe) на тему:

```
mosquitto_sub -h localhost -t "sensor/temperature"
```

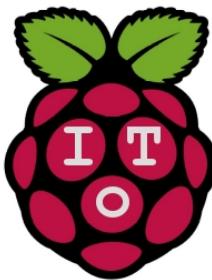
Командный файл, публикующий показания температуры со встроенного датчика Raspberry Pi:

```
#!/bin/bash
while true
do
    temperature=`cat /sys/class/thermal/thermal_zone0/temp`
    degree=$((temperature/1000))      # целая часть
    fraction=$((temperature%1000))    # дробная часть
    `mosquitto_pub -h localhost -t "sensor/temperature" -m $degree.$fraction`
    sleep 1
done
```

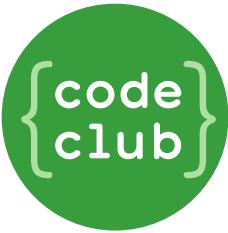
При запуске на одном компьютере следует сначала запустить брокера Mosquito, затем клиента-подписчика, затем клиента-издателя.



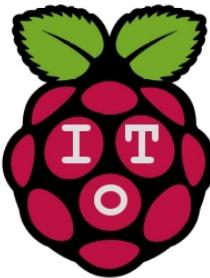
Mosquitto → Ruby



Приём из темы сообщений с показаниями температуры от брокера Mosquitto:



Mosquitto ← Ruby



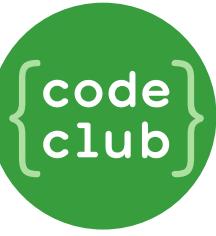
Отправка на брокер Mosquitto в заданную тему сообщений с показаниями температуры со встроенного термодатчика Raspberry Pi:

```
#!/usr/bin/ruby
require 'mqtt'                      # подключить библиотеку для работы с mqtt
require 'thermal_sensor'             # подключить библиотеку для работы с термодатчиком

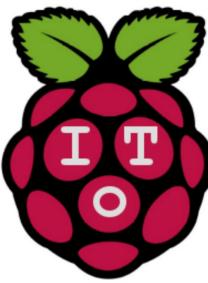
BROKER = '10.36.254.16'                # адрес или имя брокера

sensor = RaspberryPi::ThermalSensor.new # создать объект «датчик»

while true do                          # в бесконечном цикле
    sensor.read_data                  # считать показание датчика
    t = sensor.celsius.to_s           # преобразовать его в строку
    MQTT::Client.connect(BROKER) do |client| # подключиться к брокеру
        client.publish('sensor/temperature', t) # опубликовать в теме сообщение
    end
    sleep 1
end
```

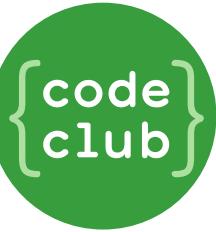


Domoticz

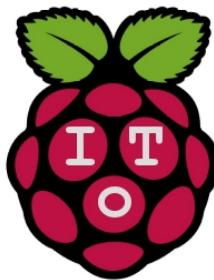


Domoticz — это сервер домашней автоматизации с открытым исходным кодом, который позволяет получать и отображать информацию об окружающих условиях (счётчики, датчики ультрафиолетового излучения, давления, влажности, температуры, протечки, открытия дверей и т. д) и управлять различными устройствами (выключатели, светильники, вентиляторы, обогреватели и т. п.). Он работает под управлением различных операционных систем, включая Raspberry Pi, имеет масштабируемый веб-интерфейс на HTML5, который автоматически адаптируется для настольных и мобильных устройств.

The screenshot displays the Domoticz web interface. On the left, a vertical list of devices and their current states (e.g., 'Выключен' - Off) is shown, along with their last seen times. The top navigation bar includes links for 'Панель' (Panel), 'План помещения' (Room Plan), 'Переключатели' (Switches), 'Температура' (Temperature), 'Погода' (Weather), 'Вспомогательные' (Auxiliary), and 'Настройка' (Settings). A dropdown menu indicates the room is 'Все' (All). Below this, a 3D floor plan of a house is shown, with various rooms and their respective device icons and status indicators. The floor plan includes labels for 'Выключен' (Off), 'Закрыто' (Closed), and numerical values like 23.6°C, 42%, 25.9°C, 41%, etc. At the bottom of the interface, there are weather and pressure indicators: 2.3°C, 42%, 1044 hPa, and 17.9°C, 22%.

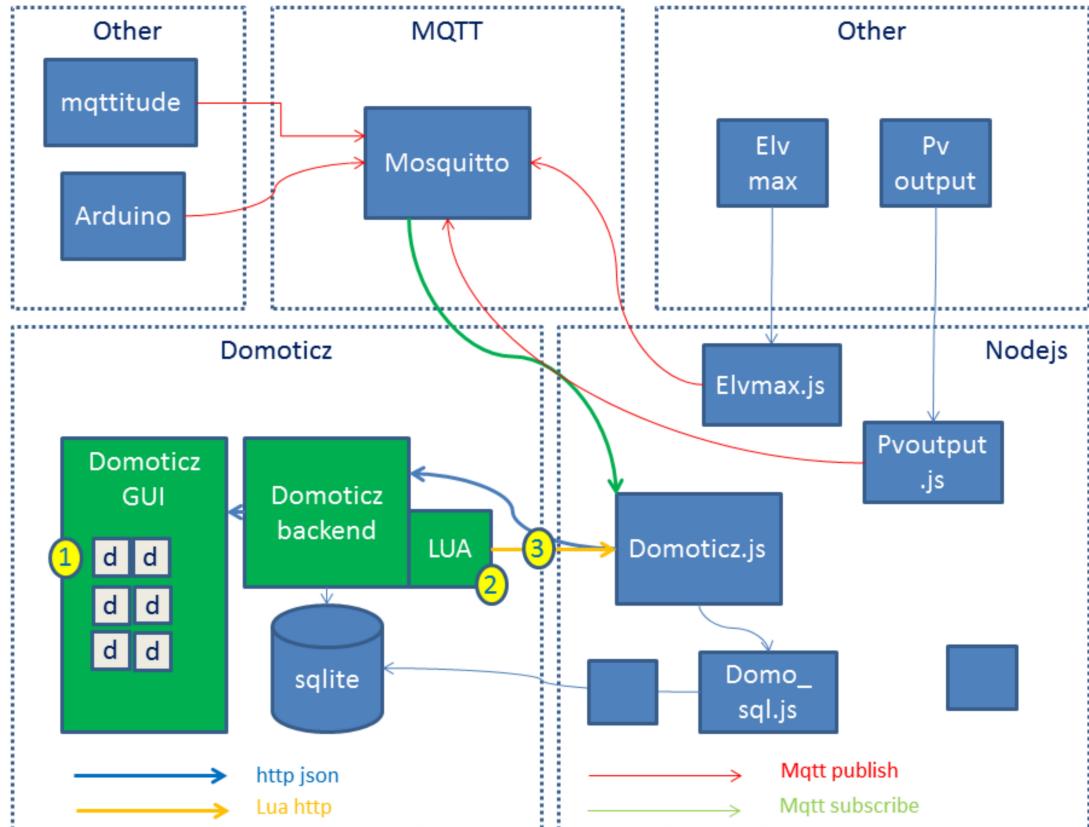


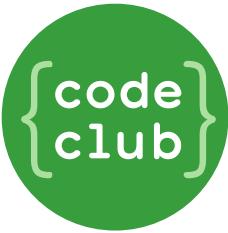
Domoticz ↔ MQTT



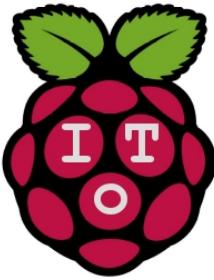
Domoticz может «общаться» со многими «умными устройствами» напрямую по «родным» протоколам (rfxtrx433, zwave, smartmeter и т. д.).

Но поскольку разнообразных устройств великое множество, и у них самые разные интерфейсы, то для них Domoticz может публиковать события посредством встроенного интерфейса MQTT, а также реагировать на действия, запрошенные внешними устройствами через брокер MQTT (например, Mosquitto).





Источники



Книги:

- Петин В. **Arduino и Raspberry Pi в проектах Internet of Things**, 2-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2018. - 432 с.: ил.

Ссылки на Интернет-ресурсы:

- Raspberry Pi:Настройка/Что_такое_протокол_MQTT_и_как_он_работает
- Платформа ARM и брокер MQTT, как современная основа решенияй для Интернета вещей
- Configuring MQTT on the Raspberry Pi
- MQTT gem для Ruby
- Программируем управление освещением по датчикам движения и освещения на Node-RED
- Domoticz на Raspberry Pi: установка, настройка, добавление первого датчика