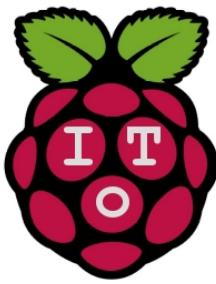


Internet of Things



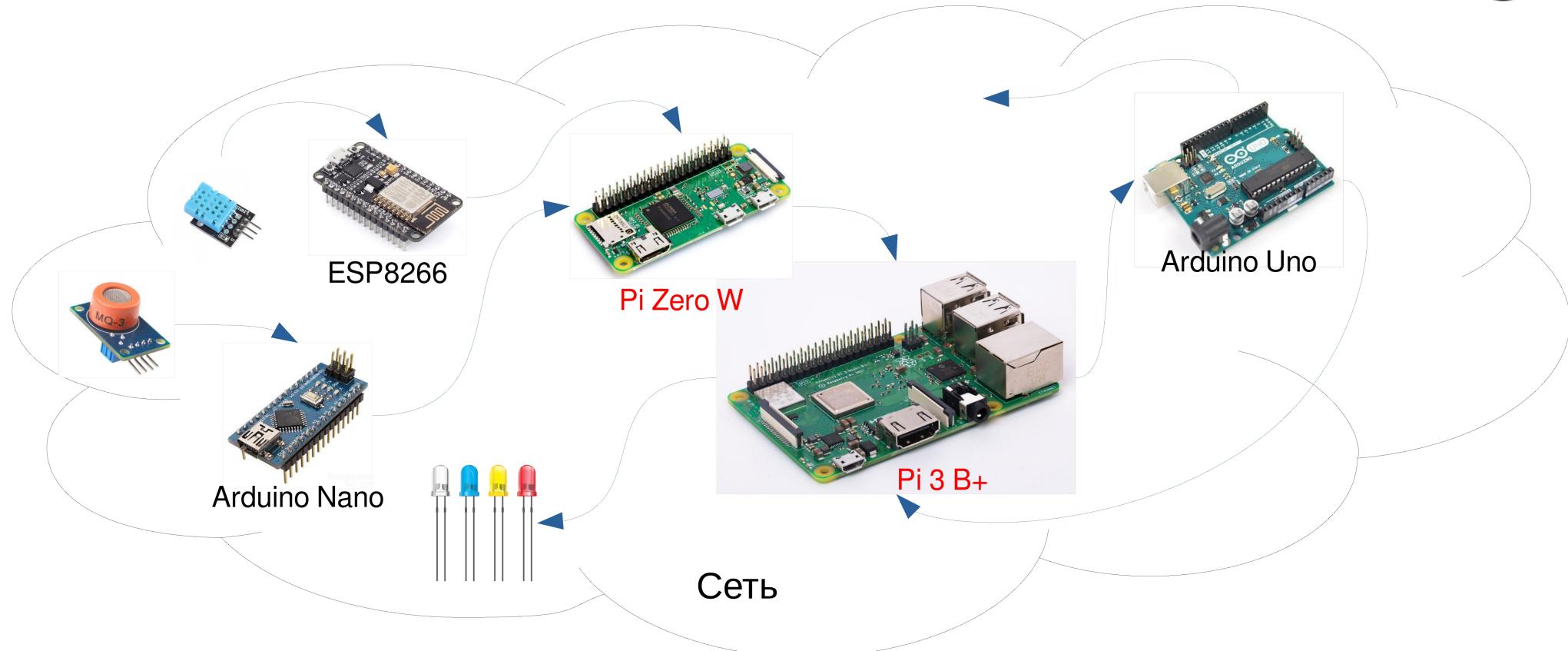
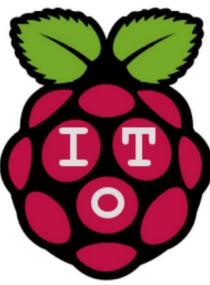
MQTT в сетях IoT

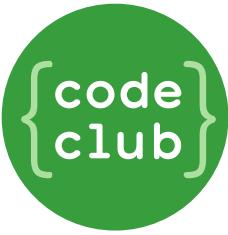
Шадринск
2018-2019

M. B. Шохирев

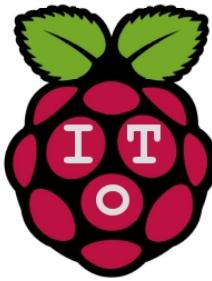
{code
club}

ИОТ = сеть контроллеров!





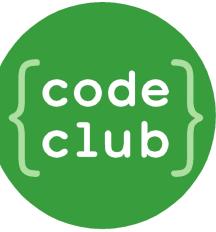
Сети IoT: всё плохо!



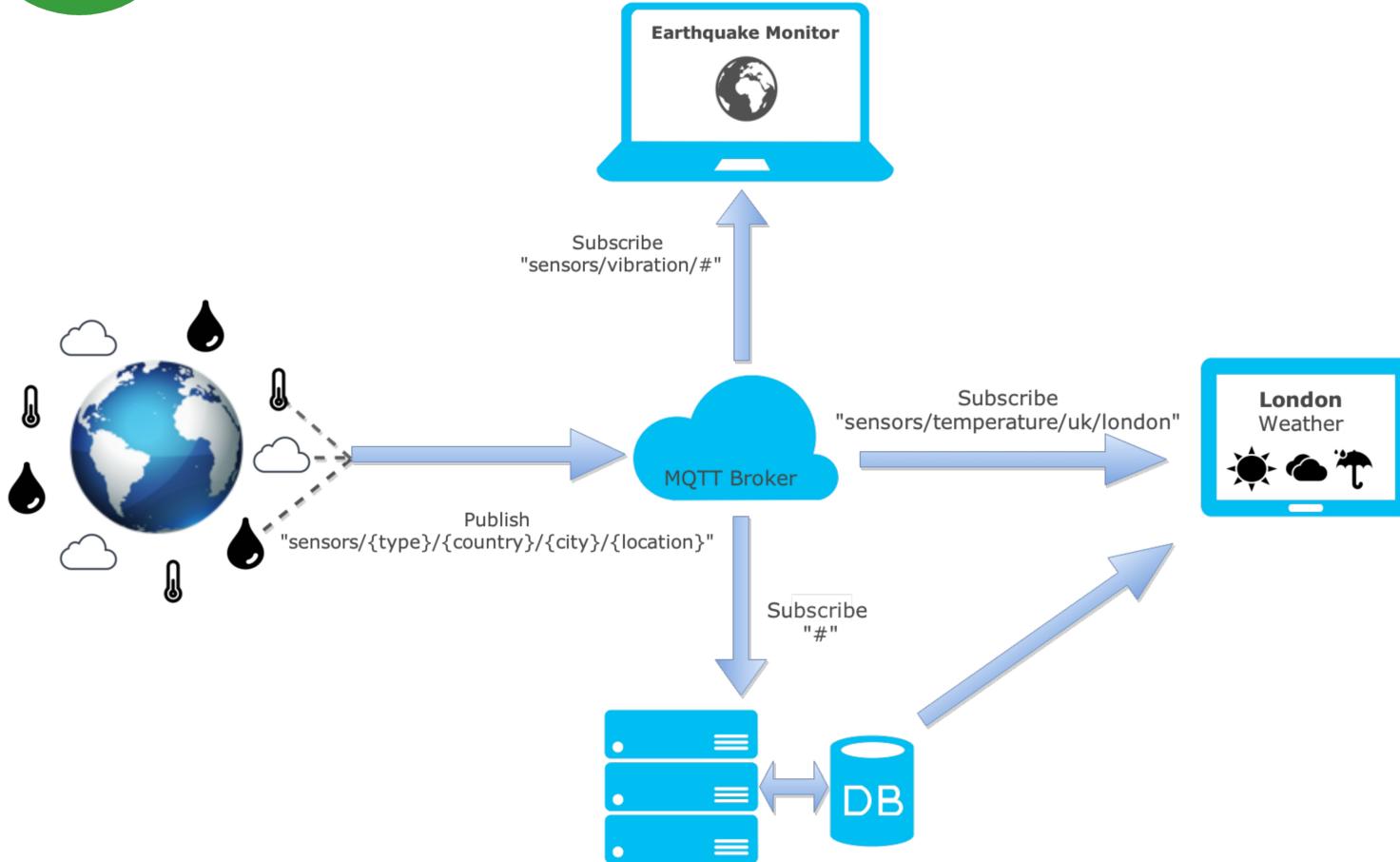
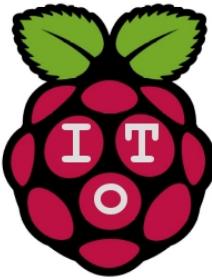
В сетях IoT типичны такие трудности:

- 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

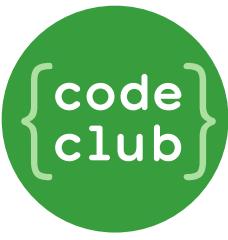
Требуется повышенная надёжность работы: исправление ошибок, повторная отправка данных, дублирование каналов передачи данных, резервирование узлов сети, самовосстановляемость систем, защита от злоумышленников.



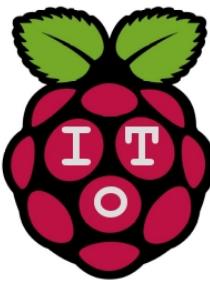
MQTT



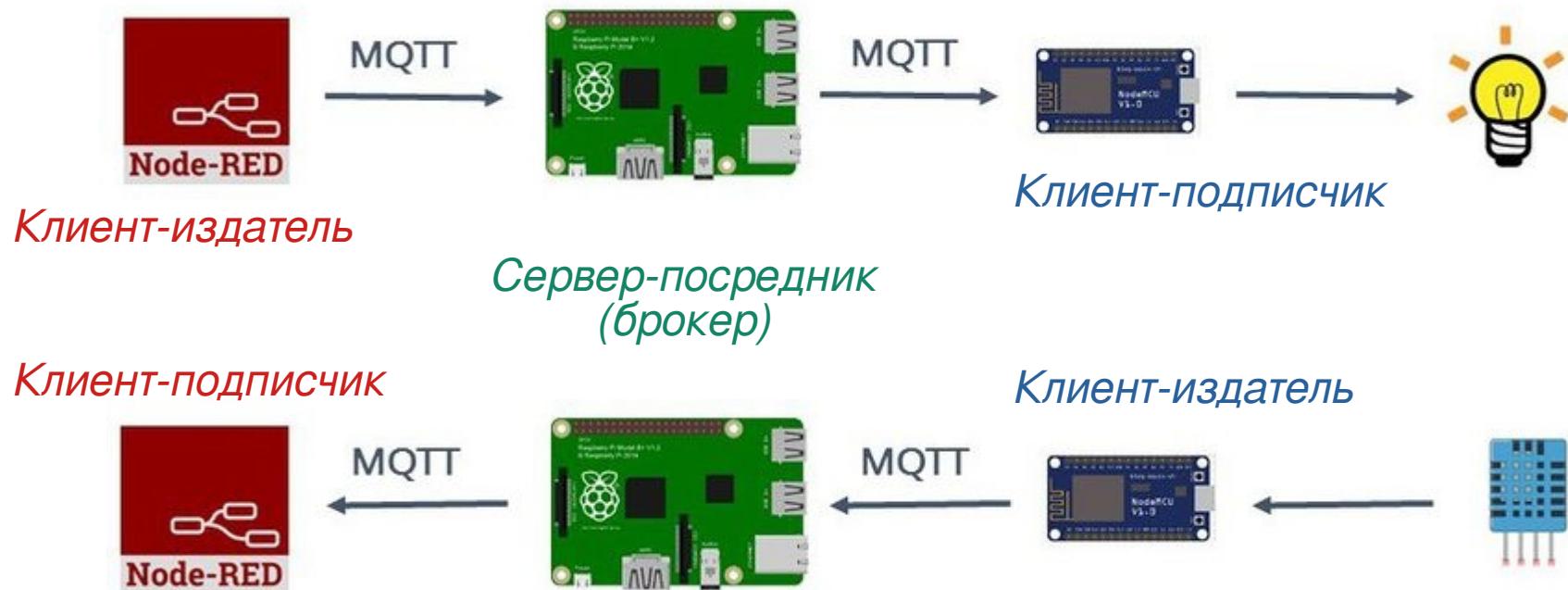
MQTT (Message Queuing Telemetry Transport) — сетевой протокол для передачи телеметрии при межмашинном взаимодействии (M2M = Machine-to-Machine). Его легко освоить и просто применять, он не сильно загружает каналы передачи данных, надёжен в работе при частых потерях связи, легко встраивается в любую систему,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

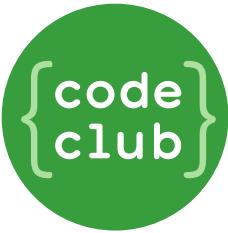


Модель MQTT

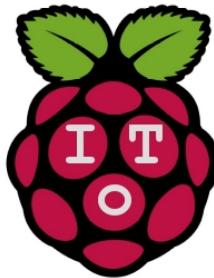


Протокол **MQTT** (Message Queuing Telemetry Transport) реализует обмен сообщениями по принципу «издатель-подписчик» с использованием посредника-брюкера.



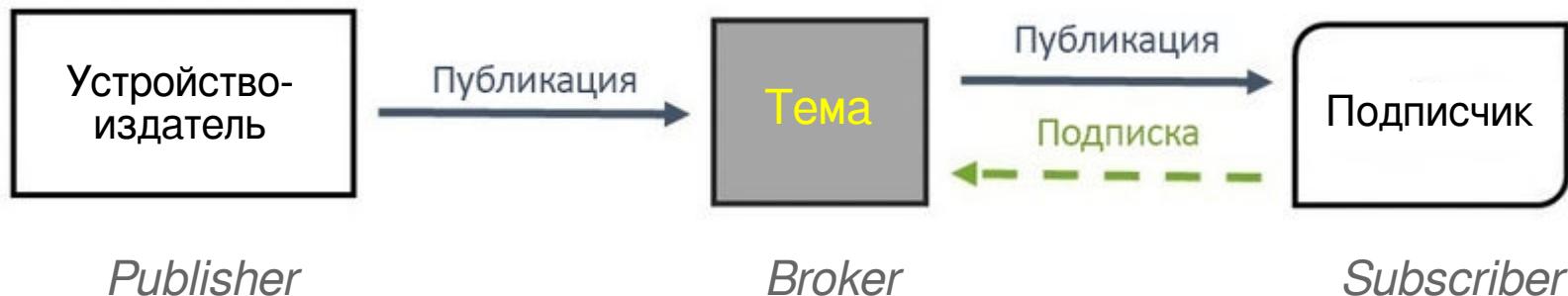


Издатель–подписчик

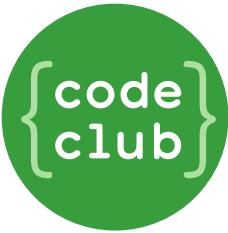


Модель «издатель–подписчик» означает:

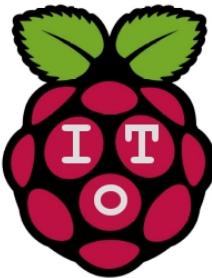
- одно устройство может публиковать (publish) сообщения для других устройств;
- устройство может подписаться (subscribe) на какую-нибудь тему (topic), чтобы получать интересующие его сообщения.



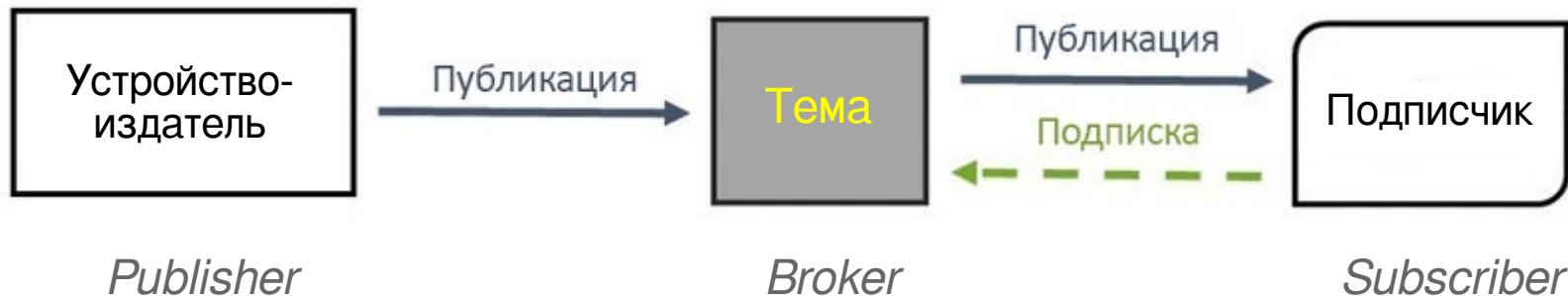
Когда одно устройство-издатель публикует сообщение в теме, а второе устройство-подписчик подписано на эту тему, то сообщение, опубликованное первым устройством, будет получено вторым устройством.



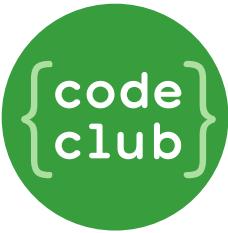
Темы



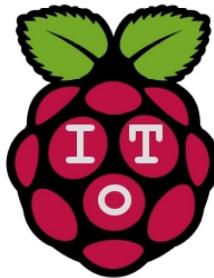
Темы (topic) — это способ группировать и структурировать сообщения, передаваемые через брокер. Имена тем и подтем разделяются символом «/». При именовании тем учитывается регистр букв (заглавные и строчные).



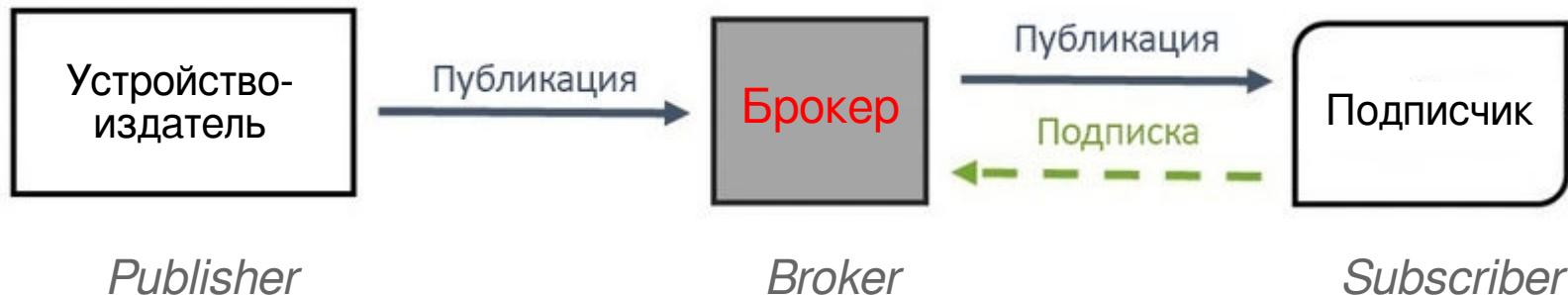
Темы могут называться строками на национальных языках, например, на русском, в кодировке UTF-8. Но рекомендуется называть темы осмысленными словами на английском языке в латинице.



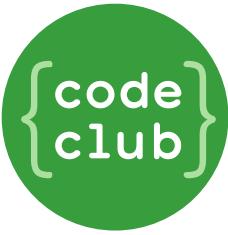
Брокер (посредник)



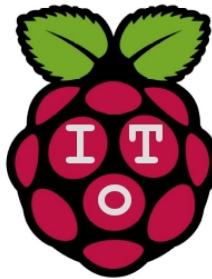
Брокер (Broker) – программа, которая организует публикацию сообщений в темах, получение сообщений от издателей, их отбор по подпискам и отправку подписчикам.



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



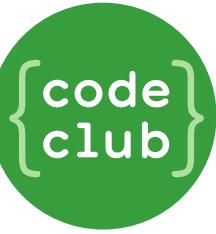
Управление по MQTT



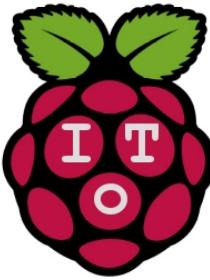
Например, чтобы включить светильник в офисе, нужно в управляющей клиентской программе, например, Node-RED на Raspberry Pi, опубликовать сообщение «ON» в теме `home/office/lamp` на брокере (который также развернут на Raspberry Pi).

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

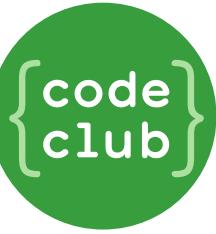


Mosquitto™ — брокер MQTT с открытым исходным кодом, развиваемый организацией Eclipse Foundation. Это легковесный сервер, который обеспечивает обмен сообщениями по протоколу MQTT. Он прекрасно подходит для обмена данными в сетях IoT между датчиками с низким энергопотреблением и мобильными устройствами,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

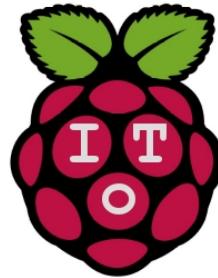
Проект **Mosquitto** также предоставляет библиотеку на языке С для реализации клиентов MQTT, а также популярные команды `mosquitto_pub` и `mosquitto_sub` для работы из командной строки.

0. Установить сервер и клиентов Mosquitto:

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



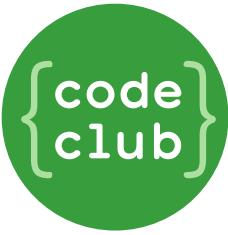
Mosquitto ← команды



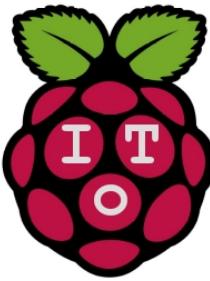
The image shows two terminal windows on a Linux desktop. The top window has a blue title bar and displays the command `mosquitto_sub -h localhost -t "sensor/temperature"`. The bottom window has a blue title bar and displays two commands: `mosquitto_pub -h localhost -t "sensor/temperature" -m 23.5` and `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



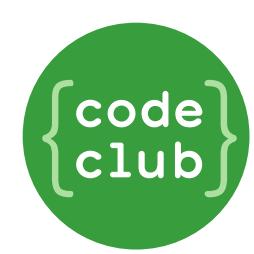
Клиент-подписчик (SUBscribe), читающий сообщения из указанной темы (**-t**):

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

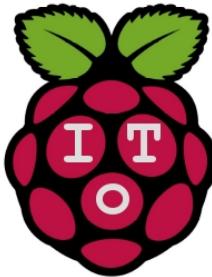
Клиент-издатель — командный файл, публикующий на определённом брокере (**-h**) в указанной теме (**-t**) сообщения (**-m**) с показаниями температуры от встроенного датчика Raspberry Pi:

```
#!/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
```

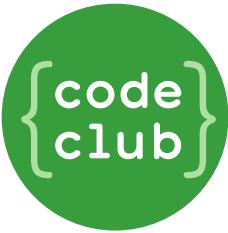
При запуске на одном компьютере следует запускать программы в следующем порядке: (0) брокер Mosquito (обычно стартует автоматически), (1) клиент-подписчик, (2) клиент-издатель.



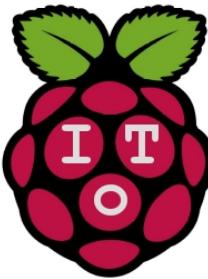
Mosquitto → Ruby



Приём из темы сообщений с показаниями температуры от брокера Mosquitto:



Mosquitto ← Ruby

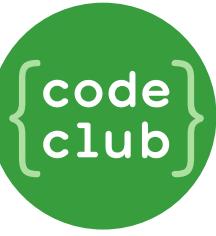


Отправка на брокер Mosquitto в заданную тему сообщений с показаниями температуры со встроенного термодатчика Raspberry Pi:

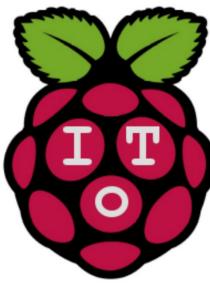
```
#!/usr/bin/ruby
require 'mqtt'                      # подключить библиотеку для работы с mqtt
require 'thermal_sensor'             # подключить библиотеку для работы с термодатчиком

BROKER = '127.0.0.1' # 'localhost'      # адрес или имя брокера
sensor = RaspberryPi::ThermalSensor.new # создать объект «датчик»

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

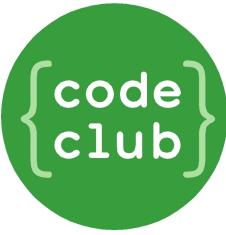


Domoticz

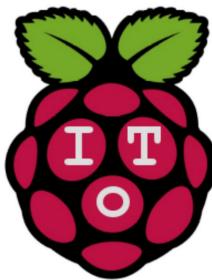


Domoticz — это свободный и открытый сервер домашней автоматизации с открытым исходным кодом, который позволяет получать и отображать информацию об окружающих условиях (счётчики, датчики ультрафиолетового излучения, давления, влажности, температуры, протечки, открытия дверей и т. д) и управлять различными 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 (Выключен or Включен) and last seen times. On the right, a 3D floor plan of a house with various sensors and actuators represented by icons and labels like 'Выключен', 'Закрыто', and temperature values (e.g., 23.6°C, 42%). A large blue circular logo with a stylized letter 'D' is positioned at the bottom center.



Domoticz ↔ MQTT

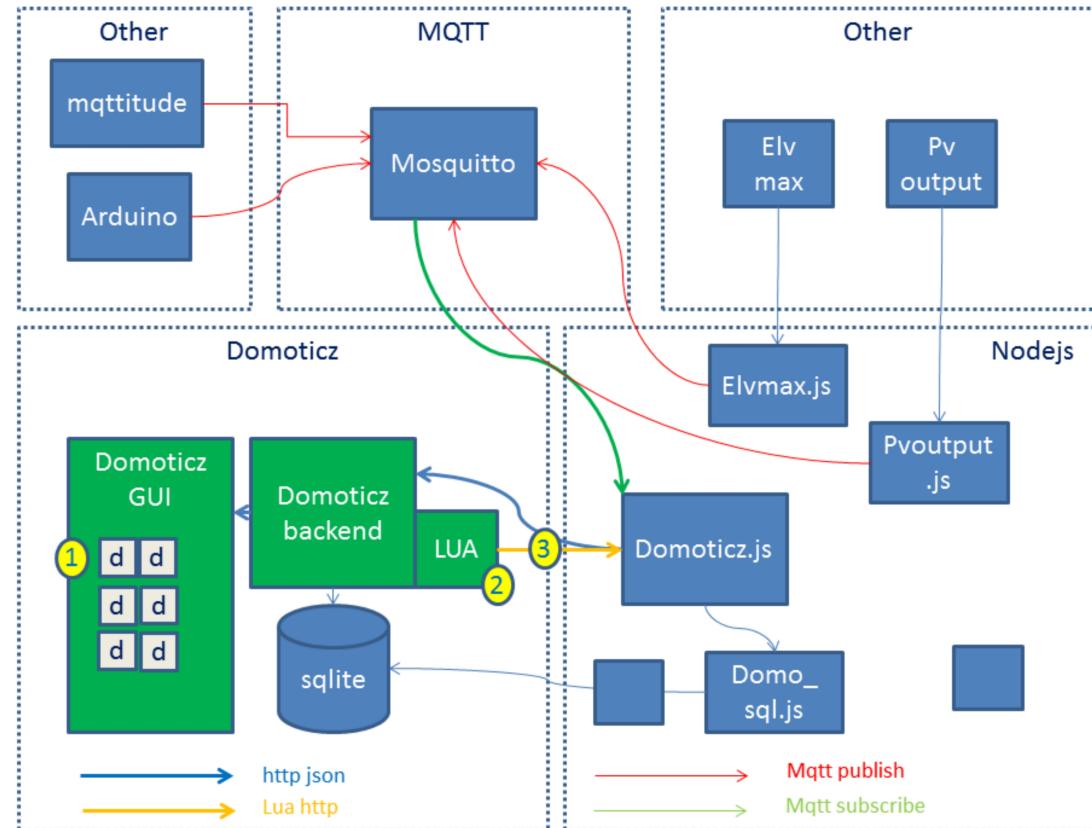


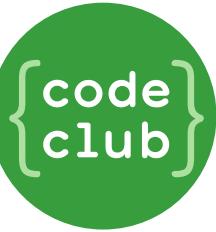
Domoticz может «общаться» со многими «умными устройствами» напрямую по «родным» протоколам (rfxtrx433, zwave, smartmeter и т. д.).

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).

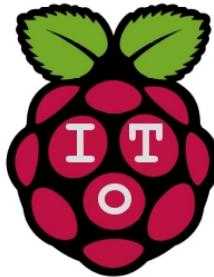
В Domoticz для приёма входящих и отправки исходящих сообщений MQTT определены 2 темы:

domoticz/in
domoticz/out





Domoticz + MQTT + JSON



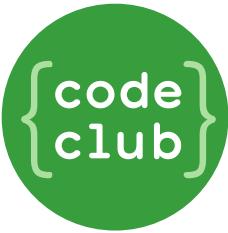
Сообщения для Domoticz оформляются в формате JSON по правилам открытого API. В этих сообщениях "idx" обозначает номер устройства, зарегистрированного на сервере Domoticz. Вот примеры исходящих (domoticz/out) и входящих (domoticz/in) сообщений:

Domoticz → MQTT (domoticz/out)

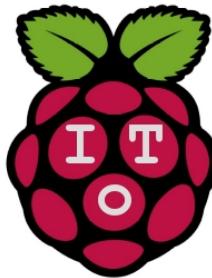
```
{  
    "Battery" : 255,  
    "RSSI" : 12,  
    "description" : "t° на плате RPi",  
    "dtype" : "Temp",  
    "id" : "1",  
    "idx" : 2,  
    "name" : "RPi Internal Temperature",  
    "nvalue" : 0,  
    "stype" : "LaCrosse TX3",  
    "svalue1" : "58.0",  
    "unit" : 1  
}
```

MQTT → Domoticz (domoticz/in)

```
{  
    "idx" : 7, "name" : "TestTemp",  
    "nvalue" : 0, "svalue" : "25.5"  
}  
  
{"command": "switchlight", "idx": 2450,  
"switchcmd": "On", "description" :  
"Включить светильник"}  
  
{"command": "switchlight", "idx": 2450,  
"switchcmd": "Set Level", "level": 100,  
"description" : "Установить уровень  
освещения"}
```



Domoticz + HTTP + JSON



По умолчанию сервер Domoticz слушает запросы по HTTP на порту 8080: отображает web-интерфейс и принимает команды управления.

Например, чтобы передать в Domoticz для устройства номер 7 значение температуры 25.5, нужно отправить команду с помощью такого запроса HTTP GET:

```
http://server:8080/json.htm?type=command&param=udevice&idx=7&nvalue=0&svalue=25.5
```

Это соответствует такому сообщению в формате JSON[^]

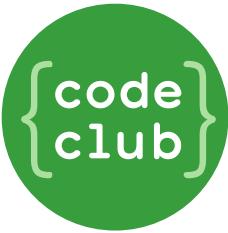
```
{ "type": "command", "param": "udevice", "idx": 7, "nvalue": 0, "svalue": 25.5 }
```

При успешном выполнении команды Domoticz ответит таким сообщением в формате JSON:

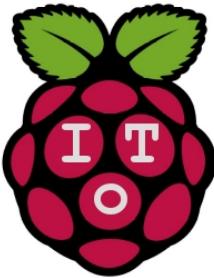
```
{ "status" : "OK", "title" : "Update Device" }
```

Если команда не выполнена (например, был указан несуществующий номер устройства), Domoticz ответит сообщением об ошибке:

```
{ "status" : "ERR" }
```



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 решений 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
- Добавляем датчик температуры DHT11/DHT22, AM2302 через GPIO в Domoticz
- <https://www.domoticz.com/wiki/MQTT>, https://www.domoticz.com/wiki/Domoticz_API/JSON_URL's