|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

**Отчет по практической работе №7**

по дисциплине «Технологические основы Интернета вещей»

|  |  |
| --- | --- |
| **Выполнил:**  Студент группыИВБО-01-22 | Воробьев Д.М. |
| **Проверил:** | Синицын Иван Васильевич |

2024 г.

СОДЕРЖАНИЕ

[1 ПРАКТИЧЕСКАЯ РАБОТА №7 – ФОРМАТЫ ПРЕДСТАВЛЕНИЯ ДАННЫХ 3](#_Toc181361028)

[1.1 Задание на практическую работу 3](#_Toc181361029)

[1.2 Код программы 3](#_Toc181361030)

[1.3 Описание работы программы 5](#_Toc181361031)

[ВЫВОД 8](#_Toc181361032)

# 1 ПРАКТИЧЕСКАЯ РАБОТА №7 – ФОРМАТЫ ПРЕДСТАВЛЕНИЯ ДАННЫХ

## 1.1 Задание на практическую работу

1. С компьютера в аудитории или личного устройства подпишитесь на несколько MQTT-топиков стенда согласно вариантам (2 вариант):
   * Датчик движения устройства WB-MSW v.3 (5);
   * Датчик освещенности устройства WB-MS v.2 (12);
   * Датчик шума устройства WB-MSW v.3 (5);
   * Датчик температуры WB-MS v.2 (12).
2. На любом языке программирования реализуйте программу (скрипт), которая бы каждые 5 секунд упаковывала последние полученные данные в файлы формата JSON и XML. В одной записи должно быть 6 полей: 4 показаний датчиков, время формирования файла, номер чемодана (последние две цифры IP-адреса).
3. На любом языке программирования реализуйте программу-парсер, которая бы выводила в консоль данные, полученные из сгенерированных в п.2 файлов.

## 1.2 Код программы

Для реализации данной практической работы был использован язык Python. Код программы представлен в Листинге 1.1.

Листинг 1.1 – Программа для подписи на стенды по MQTT

|  |
| --- |
| import xml.etree.ElementTree as ETimport paho.mqtt.client as mqtt  import jsonfrom datetime import datetime, time  from threading import Timer  HOST = "192.168.2.24" # IP адрес вашего брокера MQTTPORT = 1883 # Порт для подключения к брокеру MQTT  KEEPALIVE = 60 # Интервал keepaliveSUB\_TOPICS = {  '/devices/wb-msw-v3\_21/controls/Max Motion':  'MaxMotion', '/devices/wb-msw-v3\_21/controls/Current Motion': 'CurrentMotion', |

Продолжение Листинга 1.1

|  |
| --- |
| '/devices/wb-msw-v3\_21/controls/Sound Level': 'SoundLevel', '/devices/wb-ms\_11/controls/Illuminance': 'Illuminance',  '/devices/wb-ms\_11/controls/Temperature': 'Temperature'}  JSON\_LIST = []  JSON\_DICT = {value: 0 for value in SUB\_TOPICS.values()}JSON\_DICT['time'] = str(datetime.now())  def on\_connect(client, userdata, flags, rc):  """Обработчик события подключения к брокеру MQTT.""" print("Connected with result code " + str(rc))  for topic in SUB\_TOPICS.keys():  client.subscribe(topic)  def on\_message(client, userdata, msg): """Обработчик получения сообщений из топиков MQTT."""  payload = msg.payload.decode() topic = msg.topic  if topic in SUB\_TOPICS:  param\_name = SUB\_TOPICS[topic] JSON\_DICT[param\_name] = payload  JSON\_DICT['time'] = str(datetime.now())  JSON\_LIST.append(JSON\_DICT.copy())  print(topic + " " + payload)  # Запись данных в файл JSON (буферизация) with open('data.json', 'w') as file:  json\_string = json.dumps(JSON\_LIST, indent=4) file.write(json\_string)  def save\_to\_xml():  """Сохранение данных в XML-файл.""" root = ET.Element('records')  for record in JSON\_LIST:  record\_elem = ET.SubElement(root, 'record') for key, value in record.items():  child = ET.SubElement(record\_elem, key) child.text = str(value)  tree = ET.ElementTree(root)  tree.write('data2.xml', encoding='utf-8', xml\_declaration=True)  def periodic\_xml\_save(interval=60): """Периодическая функция для сохранения данных в XML."""  save\_to\_xml() Timer(interval, periodic\_xml\_save, [interval]).start()  def parse\_json(file\_path):  """Парсинг JSON-файла и вывод данных в консоль.""" with open(file\_path, 'r') as  json\_file:  data = json.load(json\_file) print(data)  def parse\_xml(file\_path):  """Парсинг XML-файла и вывод данных в консоль.""" tree = ET.parse(file\_path)  root = tree.getroot() for record in root.findall('record'):  data = {child.tag: child.text for child in record} print(data)  def main():  """Главная функция для подключения к брокеру и запуска обработчиков.""" client = mqtt.Client()  client.on\_connect = on\_connect client.on\_message = on\_message  client.connect(HOST, PORT, KEEPALIVE) client.loop\_start()  # Запускаем периодическое сохранение в XML с интервалом 60 секунд  periodic\_xml\_save(60)  try: while True:  pass # Основной цикл, можно добавить другие задачи except KeyboardInterrupt:  print("Завершение работы...") finally:  client.loop\_stop() client.disconnect()  if \_\_name\_\_ == "\_\_main\_\_":  main() |

## 1.3 Описание работы программы

Будут использованы импортированные библиотеки:

* + xml.etree.ElementTree - библиотека для работы с XML;
  + import paho.mqtt.client – библиотека для работы с MQTT;
  + import json – библиотека для работы с JSON;
  + from datetime import datetime, time – для обработки текущего времени.

В качестве параметров для подключения к MQTT-брокеру используются:

* + HOST - IP чемодана = 192.168.2.27 в нашем случае;
  + PORT - Стандартный порт для подключения Mosquitto = 1883;
  + KEEPALIVE - Время ожидания доставки сообщения, если при отправке оно будет превышено, брокер будет считаться недоступным = 60.

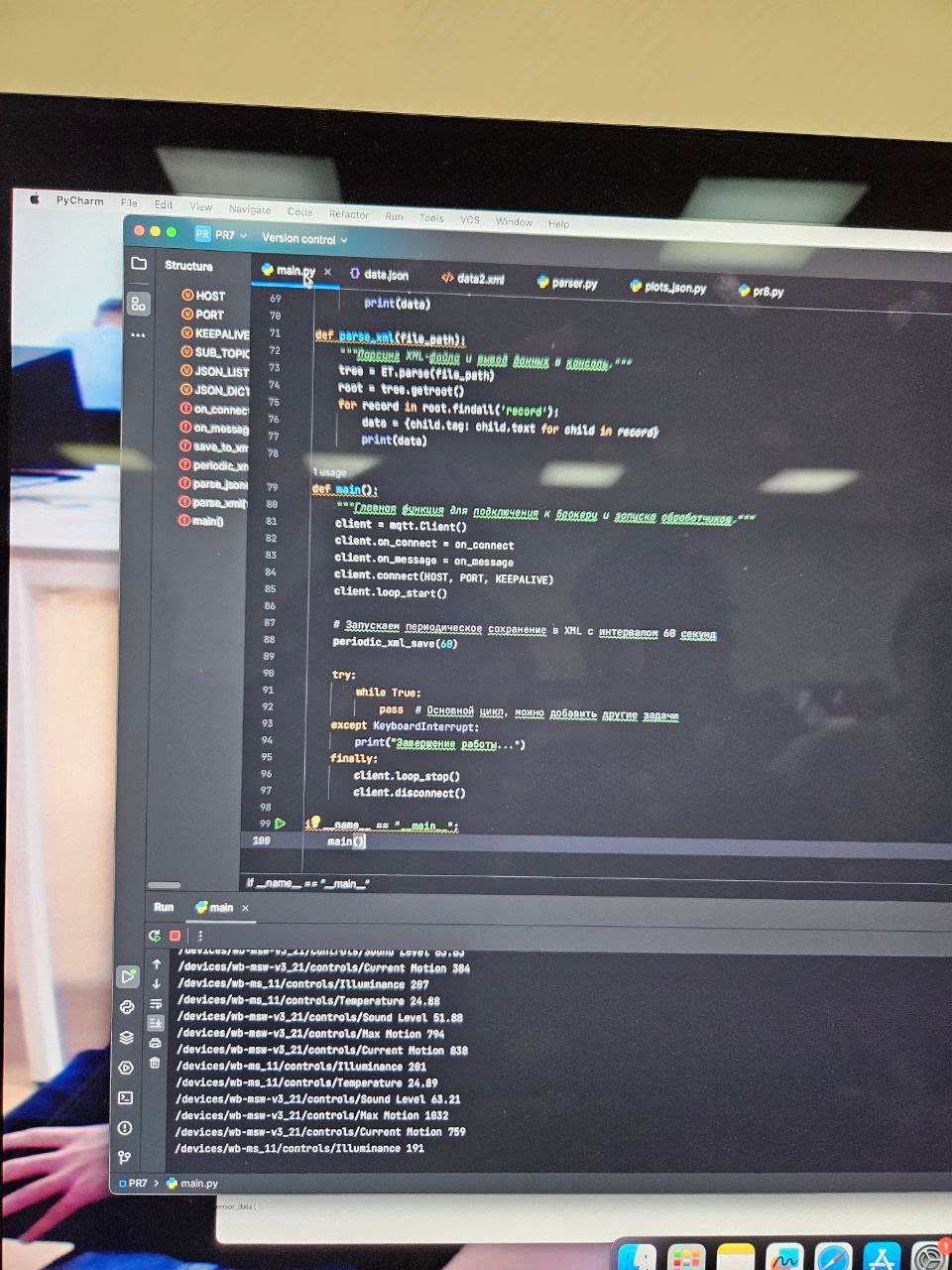
В качестве словаря с топиками использована SUB\_TOPICS, в котором приведены наименования девайсов и контроллеров, с которых будут собираться данные и в дальнейшем заноситься в .json и .xml файлы. 

Рисунок 1.1 – Устройства с которых будут собираться данные и занесенные в словарь с топиками

JSON\_LIST = [] – список для хранения данных в формате «.json».

JSON\_DICT = {} – словарь для хранения значений параметров.

def on\_connect(client, userdata, flags, rc) - метод при подключении к MQTT брокеру подписывается на все субтопики.

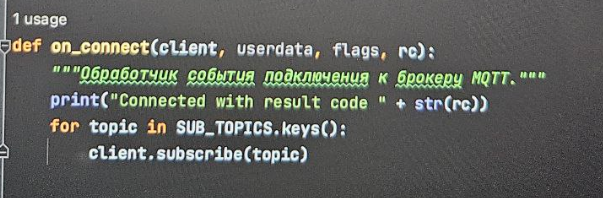


Рисунок 1.2 – Метод on\_connect

# ВЫВОД

Данная практическая работа позволила изучить основные принципы работы с протоколом MQTT, реализовать программы для сбора данных с датчиков, упаковки данных в различные форматы и последующего анализа.