

# Compte-rendus SAE24

Groupe 13

22 juin 2022

## Table des matières

<b>1</b>	<b>Réseau</b>	<b>4</b>
<b>2</b>	<b>Téléphonie</b>	<b>5</b>
<b>3</b>	<b>Collecte</b>	<b>6</b>
3.1	Connexion au broker MQTT . . . . .	6
3.2	Écriture d'un parser pour les données MQTT . . . . .	6
<b>4</b>	<b>Web</b>	<b>8</b>

## Table des figures

## Table des codes

1	Simple script imprimant les messages MQTT . . . . .	6
2	Exemple de message brut . . . . .	6
3	Séparation des valeurs . . . . .	6

# 1 Réseau

## 2 Téléphonie

## 3 Collecte

### 3.1 Connexion au broker MQTT

Pour nous connecter au Broker MQTT, nous allons utiliser le paquet Python `paho-mqtt` que nous pouvons installer avec `pip install`. Une fois le paquet installé, nous pouvons créer un simple script pour nous abonner à notre topic ainsi que d'imprimer les messages dans le terminal.

```
import paho.mqtt.client as mqtt

def on_connect(client, userdata, flags, rc):
    print("Connected with result code "+str(rc))
    client.subscribe("IUT/Colmar/SAE24/Maison1")

def on_message(client, userdata, msg):
    print(msg.topic+" "+str(msg.payload))

client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message

client.connect("test.mosquitto.org", port=1883, keepalive=60)

client.loop_forever()
```

Code 1 – Simple script imprimant les messages MQTT

On remarque que l'on crée deux fonctions `on_connect` et `on_message`, la première est exécutée lors de la connexion au broker tandis que la deuxième est exécutée lors de la réception d'un message MQTT. Nous pouvons donc utiliser ces fonctions pour nos analyses et nos opérations SQL.

### 3.2 Écriture d'un parser pour les données MQTT

Les données envoyées par les capteurs sont présentées sous un format *csv* (Comma-separated values) comme en exemple ci-dessous (Code 2).

```
Id=B8A5F3569EFF,piece=sejour,date=22/06/2022,time=21:23:53,temp=11.56
```

Code 2 – Exemple de message brut

Il est donc facile de séparer ces valeurs avec Python grâce à la méthode `split(',')`. Nous remarquons que chaque valeur est précédée de `=`. Comme nous avons pas besoin de tout ce qui est avant le signe égal, nous pouvons utiliser la même méthode que précédemment pour ne garder uniquement les valeurs utiles.

```
mac_addr= payload[0].split("=")[1]
piece= payload[1].split("=")[1]
dt= datetime.strptime(payload[2].split("=")[1] + " " + payload[3].split("=")[1],
                      '%d/%m/%Y %H:%M:%S')
temp= payload[4].split("=")[1]
```

Code 3 – Séparation des valeurs

En Code 3 nous utilisons également la méthode `strptime(str, format)` de l'objet *datetime*, qui nous permettra, en spécifiant le format de la date d'origine, d'importer la date en format ISO 8601 dans la base de données.

## 4 Web