LAPORAN PRAKTIKUM
IOT

Bowo Santoso H Manalu
11323001
D-III TEKNOLOGI INFORMASI

INSTITUT TEKNOLOGI DEL
FAKULTAS VOKASI
2024/2025

```python
import json
from topic import Topic
from data_classes import BrokerSettings, ClientSettings

```

mengimpor pustaka bawaan Python json, yang berguna untuk membaca dan menulis data dalam format JSON. Selain itu, Topic diimpor dari modul topic, yang mungkin merupakan file Python di proyek Anda. Kelas BrokerSettings dan ClientSettings diimpor dari modul data_classes, kemungkinan digunakan untuk merepresentasikan pengaturan atau konfigurasi dalam bentuk objek dengan atribut tertentu.

```python
class Simulator:
    def __init__(self, settings_file):
        self.default_client_settings = ClientSettings(
            clean=True,
            retain=False,
            qos=2,
            time_interval=10
        )
        self.topics = self.load_topics(settings_file)
```

mendefinisikan kelas Simulator dengan metode inisialisasi (__init__). Saat objek Simulator dibuat, ia menerima parameter settings_file. Di dalamnya, atribut default_client_settings diatur sebagai objek ClientSettings dengan beberapa nilai default seperti clean=True, retain=False, qos=2, dan time_interval=10. Selain itu, atribut topics diisi dengan hasil pemanggilan metode load_topics, yang memuat topik dari file pengaturan yang diberikan.

```python
def read_client_settings(self, settings_dict: dict, default: ClientSettings):
    return ClientSettings(
        clean=settings_dict.get('CLEAN_SESSION', default.clean),
        retain=settings_dict.get('RETAIN', default.retain),
        qos=settings_dict.get('QOS', default.qos),
        time_interval= settings_dict.get('TIME_INTERVAL', default.time_interval)
    )
```

metode bernama read_client_settings yang menerima dua parameter: settings_dict (sebuah dictionary) dan default (objek ClientSettings). Metode ini mengembalikan objek ClientSettings baru, di mana setiap atributnya diambil dari settings_dict jika tersedia. Jika tidak, nilai default dari objek default akan digunakan. Contohnya, nilai untuk clean diambil dari CLEAN_SESSION di dictionary, atau dari default.clean jika kunci tersebut tidak ada.

```python
def load_topics(self, settings_file):
    topics = []
    with open(settings_file) as json_file:
        config = json.load(json_file)
        broker_settings = BrokerSettings(
            url=config.get('BROKER_URL', 'localhost'),
            port=config.get('BROKER_PORT', 1883),
            protocol=config.get('PROTOCOL_VERSION', 4) # mqtt.MQTTv311
        )
        broker_client_settings = self.read_client_settings(config, default=self.default_client_settings)
        # read each configured topic
        for topic in config['TOPICS']:
            topic_data = topic['DATA']
            topic_payload_root = topic.get('PAYLOAD_ROOT', {})
            topic_client_settings = self.read_client_settings(topic, default=broker_client_settings)
            if topic['TYPE'] == 'single':
                # create single topic with format: /{PREFIX}
                topic_url = topic['PREFIX']
                topics.append(Topic(broker_settings, topic_url, topic_data, topic_payload_root, topic_client_settings))
            elif topic['TYPE'] == 'multiple':
                # create multiple topics with format: /{PREFIX}/{id}
                for id in range(topic['RANGE_START'], topic['RANGE_END']+1):
                    topic_url = topic['PREFIX'] + '/' + str(id)
                    topics.append(Topic(broker_settings, topic_url, topic_data, topic_payload_root, topic_client_settings))
            elif topic['TYPE'] == 'list':
                # create multiple topics with format: /{PREFIX}/{item}
                for item in topic['LIST']:
                    topic_url = topic['PREFIX'] + '/' + str(item)
                    topics.append(Topic(broker_settings, topic_url, topic_data, topic_payload_root, topic_client_settings))
    return topics
```

metode load_topics yang memuat daftar topik dari file pengaturan yang diberikan. File dibuka dan dibaca sebagai JSON untuk mendapatkan konfigurasi. Kemudian, objek BrokerSettings dibuat menggunakan nilai dari file JSON, atau nilai default jika kunci tertentu tidak ada. Pengaturan klien broker dibuat dengan memanggil metode read_client_settings.

Setelah itu, metode ini membaca setiap topik yang dikonfigurasi di dalam file. Jika tipe topik adalah single, dibuat satu topik dengan format URL berdasarkan PREFIX. Jika tipe topik adalah multiple, dibuat beberapa topik dengan format URL yang mencakup rentang ID dari RANGE_START hingga RANGE_END. Jika tipe topik adalah list, dibuat beberapa topik berdasarkan daftar item yang diberikan di LIST. Semua topik yang dibuat ditambahkan ke daftar topics, dan daftar ini dikembalikan sebagai hasil metode.

```python
def run(self):
    for topic in self.topics:
        print(f'Starting: {topic.topic_url} ...')
        topic.start()
    for topic in self.topics:
        # workaround for Python 3.12
        topic.join()
```

metode run yang menjalankan semua topik yang ada dalam daftar self.topics. Untuk setiap topik, ia mencetak pesan "Starting" beserta URL topik, lalu memulai proses topik dengan memanggil metode start. Setelah itu, untuk setiap topik, metode join dipanggil untuk memastikan semua proses selesai sebelum melanjutkan

```
def stop(self):
    for topic in self.topics:
        print(f'Stopping: {topic.topic_url} ...')
        topic.stop()
```

metode stop yang menghentikan semua topik dalam daftar self.topics. Untuk setiap topik, ia mencetak pesan "Stopping" beserta URL topik, lalu memanggil metode stop pada topik tersebut.