강원혁신플랫폼

리눅스프로그래밍

SocketIO

웹 브라우저와 웹 서버 간의 실시간 양방향 통신을 가능하게 하는 JavaScript 라이브러리로 실시간 응용 프로그램을 구축하는데 사용하는 라이브러리는 무엇인가요?

Socket.IO

학습목표

학습 내용

1. SocketIO 패키지
2. SocketIO 응용

학습 목표

- SocketIO 패키지에 대해 설명할 수 있다.
- SocketIO를 응용할 수 있다.

SocketIO 패키지

Socket.IO

웹 브라우저와 웹 서버 간의 실시간 양방향 통신을 가능하게 하는
JavaScript 라이브러리
실시간 응용 프로그램을 구축 사용

SocketIO

주요 기능

실시간 양방향 통신

클라이언트가 주기적인 요청을 할 필요 없이
즉시 데이터를 송수신 가능

SocketIO

특징

01 • WebSockets

Socket.IO는 효율적이고 대기 시간이 짧은 전이중 통신을 위해
브라우저에서 사용 가능한 경우 WebSocket을 사용

02 • 이벤트 기반 통신

클라이언트와 서버는 이벤트를 내보내고 수신할 수 있으므로
사용자 지정 작업 및 데이터 교환을 쉽게 구현

특징

03 • 브로드캐스팅

Socket.IO는 여러 클라이언트 또는 특정 룸이나 네임스페이스에 연결된 모든 클라이언트에 메시지를 브로드캐스트

04 • 교차 플랫폼 호환성

Socket.IO는 웹 브라우저, 모바일 장치 및 서버를 포함한 다양한 플랫폼에서 작동하므로 다중 장치 지원을 통해 실시간 응용 프로그램을 구축하는 데 적합

SocketIO 패키지

새 가상환경 만들기

```
$ python -m venv flask
$ sudo apt install python3-virtualenv
$ virtualenv flask
$ source flask/bin/activate

(flask) $ sudo apt-get update
(flask) $ sudo apt-get install python3 python3-pip
(flask) $ pip3 install Flask Flask-SocketIO eventlet
```

SocketIO 테스트


```
(.venv) $ cat websocket_test.py
from flask import Flask, render_template
from flask_socketio import SocketIO

app = Flask(__name__)
socketio = SocketIO(app , cors_allowed_origins="*")

@app.route('/')
def index():
    return render_template('index_websocket.html')
```

SocketIO 테스트


```
@socketio.on('message')
def handle_message(message):
    print('Received message:', message)
    socketio.send('Server received: ' + message)

if __name__ == '__main__':
    socketio.run(app, host='0.0.0.0', port=5000)
```

SocketIO 테스트


```
(.venv) $ cat templates/index_websocket.html
/* 상단 헤더 생략 */
    <input type="text" id="message" placeholder="Type a message">
    <button onclick="sendMessage()">Send</button>
    <div id="output"></div>

    <script
src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/4.1.2/socket.io.js"></script>
    <script type="text/javascript">
        const socket = io.connect('http://localhost:5000');
```

```
    socket.on('message', function (message) {
        document.getElementById('output').innerHTML += '<p>' + message + '</p>';
    });

    function sendMessage() {
        const input = document.getElementById('message');
        const message = input.value;
        input.value = '';
        socket.send(message);
    }
</script>
```

SocketIO 테스트 실행결과


```
(.venv) $ python3 websocket_test.py
WebSocket transport not available. Install simple-websocket for improved
performance.
 * Serving Flask app 'websocket_test'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.18.93.102:5000
Press CTRL+C to quit
The WebSocket transport is not available, you must install a WebSocket server that is
```

SocketIO 테스트 실행결과

compatible with your async mode to enable it. See the documentation for details. (further occurrences of this error will be logged with level INFO)

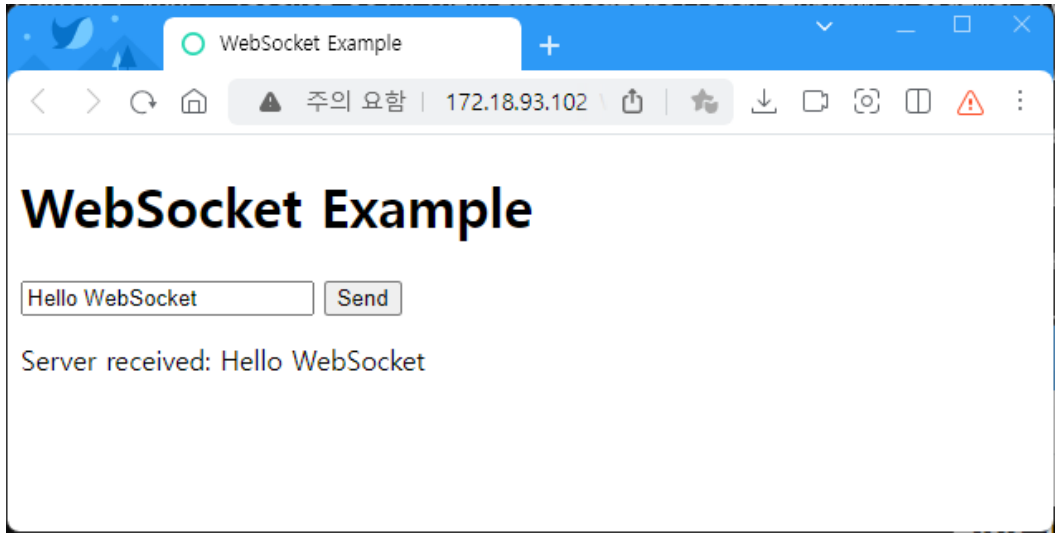
127.0.0.1 - - [15/Jul/2023 12:47:19] "GET /socket.io/?EIO=4&transport=polling&t=ObNgF50 HTTP/1.1" 200 -

127.0.0.1 - - [15/Jul/2023 12:47:19] "POST /socket.io/?EIO=4&transport=polling&t=ObNgFbq&sid=g7NGW6wWQxNbaFB2AAAA HTTP/1.1" 200 -

Received message: Hello WebSocket

127.0.0.1 - - [15/Jul/2023 12:47:46] "POST /socket.io/?EIO=4&transport=polling&t=ObNgMEq&sid=wsEofVZyRx-gTNfpAAAI HTTP/1.1" 200 -

SocketIO 테스트 실행결과



SocketIO 응용

SocketIO 응용, 패키지


```
(flask) $ pip install Flask
(flask) $ pip install pandas
(flask) $ pip install matplotlib
(flask) $ pip install flask-socketio
(flask) $ pip install pymysql
```

```
from flask import Flask, render_template
from flask_socketio import SocketIO, emit
import pymysql
import pandas as pd

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from pandas import DataFrame
from random import randrange
```

```
app = Flask(__name__)
socketio = SocketIO(app, cors_allowed_origins="*")

# Configure MySQL connection
host='localhost'
user='scott'
password='tiger'
database='mydb'
```

```
def get_sensor_data():
    conn = pymysql.connect(host=host, user=user, password=password,
database=database)
    query = "SELECT timestamp, temperature, humidity, illuminance FROM ( SELECT *
FROM SensorData ORDER BY id DESC LIMIT 100)Var1 ORDER BY id ASC"
    df = pd.read_sql(query, conn)
    df = df.set_index('timestamp')
    conn.close()
    #print(df)
    return df
```

```
def generate_plot(df):
    return df.plot(use_index=True, y=["temperature", "humidity", "illuminance"],
        kind="line", figsize=(10, 5)).legend(loc='upper left')

@socketio.on('connect')
def handle_connect():
    print('Client connected')
    emit('update_plot', 'Connected')  # Send a message to the client on connect
```

SocketIO 응용3

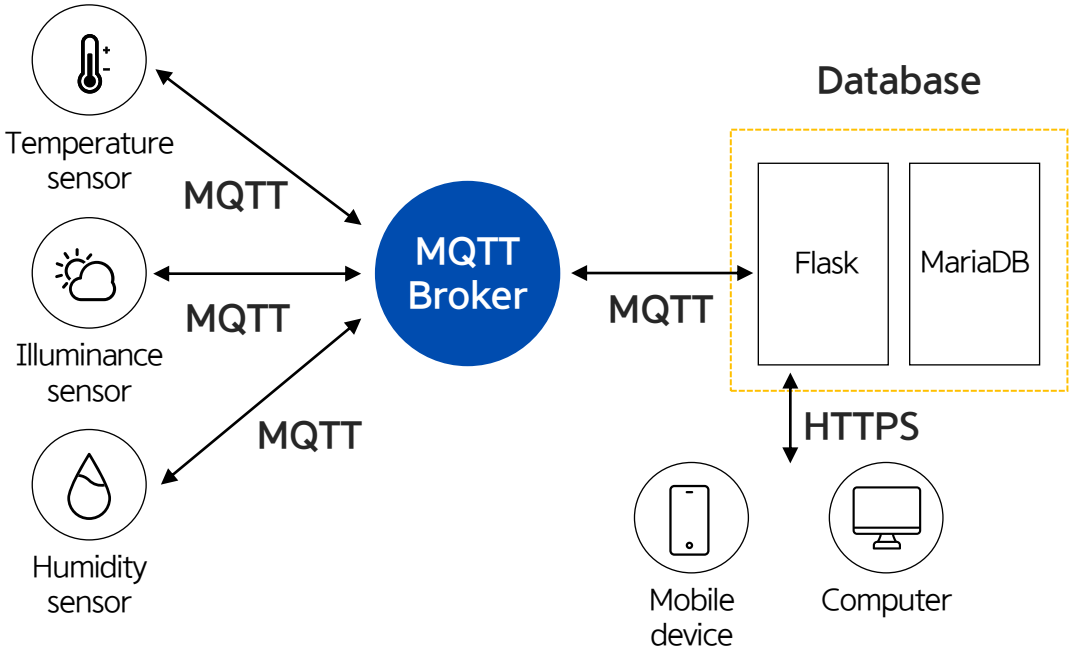

```
@app.route('/')
def index():
    return render_template('index_plot.html')

@socketio.on('get_plot')
def handle_get_plot():
    sensor_data = get_sensor_data()
    plot = generate_plot(sensor_data)
    plt.savefig('static/plot.png')  # Save the plot as an image
    plt.close()  # Close the plot
    emit('update_plot', 'plot.png')  # Send the updated plot filename to the client
```

SocketIO 응용3


```
if __name__ == '__main__':
    socketio.run(app, port=5001)
```

동작 구조

동작 구조


Temperature sensor
MQTT
Illuminance sensor
MQTT
Humidity sensor
MQTT
MQTT Broker
MQTT
Database
Flask
MariaDB
HTTPS
Mobile device
Computer

실행결과


```
(flask) $ export FLASK_APP=simul_sensors_mqtt_db.py
(flask) $ flask run

(flask) $ export FLASK_APP=helloflask_mqtt_db
(flask) $ flask run

(flask) $ python websocket_plot.py
```

Sensor Data

127.0.0.1 Sensor Data

Sensor Data

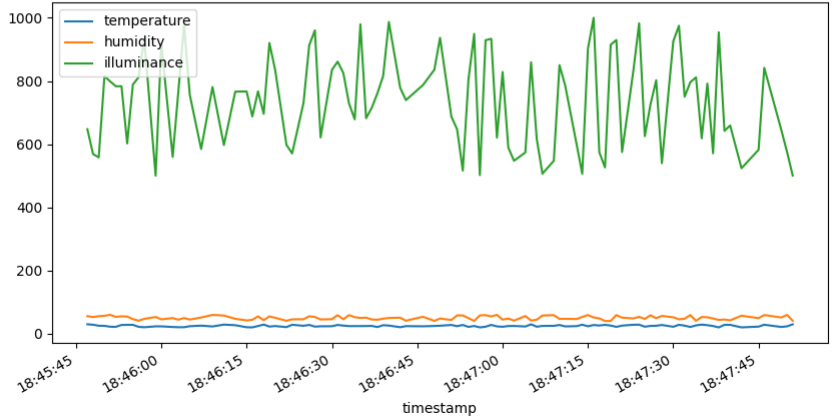
ID	Sensor ID	Reading	Timestamp	temperature	humidity	illuminance
182	3	26.0735	2023-07-14 21:06:12	25.6926	50.6736	747.93
181	2	22.0484	2023-07-14 21:06:11	20.0582	53.827	712.009
180	2	23.6572	2023-07-14 21:06:10	28.7955	42.8181	762.437
178	2	26.3963	2023-07-14 21:06:08	25.3896	41.2812	525.336
177	2	26.2081	2023-07-14 21:06:07	23.4174	44.0302	561.897
176	3	20.2233	2023-07-14 21:05:21	26.9862	55.7639	567.944
175	3	24.4537	2023-07-14 21:05:20	27.9223	51.9219	833.0
174	3	28.9594	2023-07-14 21:05:19	20.5378	45.358	882.033
173	2	25.4197	2023-07-14 21:05:18	27.98	56.2879	623.826
172	2	21.8065	2023-07-14 21:05:17	26.4346	41.6798	794.154
171	3	23.6833	2023-07-14 21:05:16	29.3001	42.2292	631.159
169	2	22.0533	2023-07-14 21:04:06	24.115	52.8927	544.658
168	2	25.0026	2023-07-14 21:04:05	24.6328	51.7361	760.95
167	3	20.8132	2023-07-14 21:04:04	24.7321	44.9524	652.604
166	2	22.7838	2023-07-14 21:04:03	28.0022	48.988	814.179
165	1	20.7743	2023-07-14 21:04:02	24.3989	48.9163	869.337
164	3	24.6654	2023-07-14 21:04:01	20.2254	45.6599	970.455
163	3	21.8512	2023-07-14 21:02:36	29.3089	48.2347	851.022

Sensor Data Plot

127.0.0.1 Sensor Data Plot

Sensor Data Plot

Update Plot



요약정리

01 • SocketIO 패키지

02 • SocketIO 응용