Для выполнения данной задачи вам будет необходимо реализовать Java-класс, который будет использовать `RestTemplate` для отправки POST-запросов на создание сенсора, а затем отправит 1000 запросов с случайными температурами и значениями "дождя". После этого он выполнит GET-запрос, чтобы получить все измерения с сервера. Напоследок, для необязательного задания с графиком, можно использовать библиотеку XChart для построения графика температур.

### Пример реализации

java

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import org.springframework.web.client.RestTemplate;

import java.util.Random;

@Service

public class SensorService {

private final RestTemplate restTemplate;

private final String baseUrl = "http://localhost:8080"; // Замените на URL вашего API

@Autowired

public SensorService(RestTemplate restTemplate) {

this.restTemplate = restTemplate;

}

public void registerSensor(String sensorName) {

String url = baseUrl + "/sensors/register";

Sensor sensor = new Sensor(sensorName);

restTemplate.postForObject(url, sensor, Sensor.class);

}

public void sendMeasurements(int count) {

Random random = new Random();

String url = baseUrl + "/measurements/add";

for (int i = 0; i < count; i++) {

Measurement measurement = new Measurement();

measurement.setTemperature(random.nextDouble() \* 100); // Случайная температура от 0 до 100

measurement.setRain(random.nextBoolean()); // Случайное значение "дождь" (true/false)

restTemplate.postForObject(url, measurement, Measurement.class);

}

}

public Measurement[] getMeasurements() {

String url = baseUrl + "/measurements";

return restTemplate.getForObject(url, Measurement[].class);

}

}

class Sensor {

private String name;

public Sensor(String name) {

this.name = name;

}

// Геттеры и сеттеры

}

class Measurement {

private double temperature;

private boolean rain;

// Геттеры и сеттеры

}

### Основной класс

Теперь создайте основной класс, который будет использовать `SensorService`:

java

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.web.client.RestTemplate;

@SpringBootApplication

public class Application implements CommandLineRunner {

public static void main(String[] args) {

SpringApplication.run(Application.class, args);

}

@Override

public void run(String... args) {

SensorService sensorService = new SensorService(new RestTemplate());

// Регистрация сенсора

sensorService.registerSensor("MySensor");

// Отправка 1000 измерений

sensorService.sendMeasurements(1000);

// Получение измерений

Measurement[] measurements = sensorService.getMeasurements();

System.out.println("Получено " + measurements.length + " измерений.");

// Добавьте код для построения графика, если необходимо

}

}

### Построение графика с использованием XChart

Если вы хотите реализовать построение графика с использованием XChart, добавьте зависимость в ваш `pom.xml`:

xml

<dependency>

<groupId>org.knowm.xchart</groupId>

<artifactId>xchart</artifactId>

<version>3.8.0</version>

</dependency>

И добавьте метод для построения графика в `SensorService`:

java

import org.knowm.xchart.LineChart;

import org.knowm.xchart.SwingWrapper;

import org.knowm.xchart.XYChart;

public void plotTemperatureChart(Measurement[] measurements) {

double[] temperatures = new double[measurements.length];

for (int i = 0; i < measurements.length; i++) {

temperatures[i] = measurements[i].getTemperature();

}

LineChart chart = new LineChart(800, 600);

chart.setTitle("Temperatures Over Time");

chart.setXAxisTitle("Measurement Number");

chart.setYAxisTitle("Temperature (°C)");

chart.addSeries("Temperatures", temperatures);

new SwingWrapper(chart).displayChart();

}

### Завершение

Не забудьте заменить базовый URL на соответствующий вашему проекту. После выполнения вашего Java-клиента и регистрации сенсора, он должен отправить 1000 измерений и вывести их на экран, а также построить график, если вы реализуете данный функционал.