RESTTemplate사용 JSON데이터 받기

1. 의존성 추가

```xml
<dependency>
    <groupId>com.fasterxml.jackson.dataformat</groupId>
    <artifactId>jackson-dataformat-xml</artifactId>
</dependency>

<dependency>
    <groupId>org.apache.httpcomponents</groupId>
    <artifactId>httpclient</artifactId>
    <version>4.5.9</version>
</dependency>
```

Json형태의 데이터를 다루기 위한 jackson관련 의존성,
RESTTemplate 및 http관련 자원을 사용하기 위한 httpclient 의존성 추가

2. http관련 설정하기

```java
package com.multicampus.finalproject.configs.resttemplate;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.http.client.HttpComponentsClientHttpRequestFactory;
import org.springframework.web.client.RestTemplate;
import org.apache.http.client.HttpClient;
import org.apache.http.impl.client.HttpClientBuilder;
@Configuration
public class HttpConnectionConfig{

    @Bean
    public RestTemplate getCustomRestTemplate(){
        HttpComponentsClientHttpRequestFactory httpRequestFactory = new HttpComponentsClientHttpRequestFactory();
        httpRequestFactory.setConnectTimeout(10000000);
        httpRequestFactory.setReadTimeout(10000000);
        //connetPool 설정
        HttpClient httpClient = HttpClientBuilder.create()
                                //connectPool 갯수
                                .setMaxConnTotal(200)
                                //ip,port 하나당 연결 제한 갯수
                                .setMaxConnPerRoute(20)
                                .build();

        httpRequestFactory.setHttpClient(httpClient);
        return new RestTemplate(httpRequestFactory);
    }
}
```

HttpConnectionConfig 자바 설정파일을 추가해준다.
이는 외부 연결에 대한 설정을 다루는 파일이다.
현재 연결시간, 읽기시간, 커넥션 수 등을 설정해준 모습이다.

3. RestTemplate사용을 위한 준비

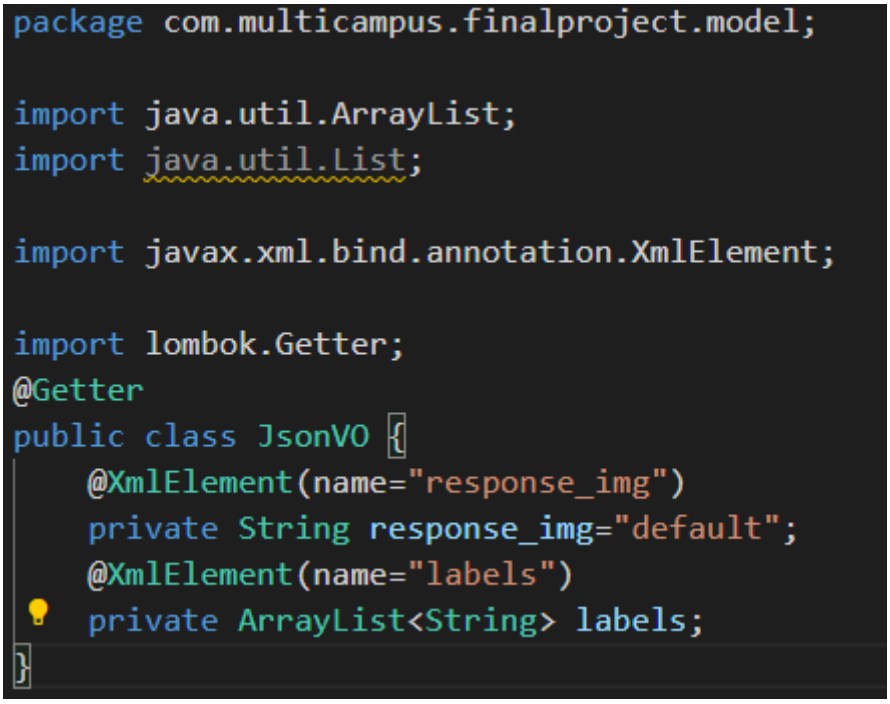
```java
import com.multicampus.finalproject.util.RestTemplateUtil;

import lombok.extern.slf4j.Slf4j;

import java.util.ArrayList;

import com.multicampus.finalproject.model.JsonVO;
import com.multicampus.finalproject.model.LabelJsonVO;
@Service
//@Slf4j
public class RestTemplateService {

    // public XmlVo getXmlData() {
    //     return RestTemplateUtil.getXmlResponse();
    // }


    // public ResponseEntity<String> getEntity(String key) {
    //     return RestTemplateUtil.getResponseEntity(key);
    // }

    public ResponseEntity<JsonVO> addData(String imgString) {
        return RestTemplateUtil.post(imgString);
    }
    public ResponseEntity<LabelJsonVO> getRecomandData(ArrayList<String> label) {
        return RestTemplateUtil.postRecomandJsonRsponse(label);
    }

}
```

이와 같은 RestTemplate를 사용하기 위한 Service를 작성해준다.
이와 연결되는 RestTemplateUtil 또한 작성해준다.
RestTemplateUtil에는 직접적으로 restTemplate를 사용하는 부분이다.

```java
public static ResponseEntity<JsonVO> post(String imgString){
    // return restTemplate.postForEntity("http://localhost:5000/testapi",imgString, JsonVO.class);
```

Json형태로 데이터를 응답받기 위한 형태이다.
restTemplate. 을 입력후 사용가능한 메소드를 확인해보면 좋다.
메소드명이 상당히 직관적이기 때문에 한눈에 알아들을 수 있을것이다.

현재는 localhost의 5000번 포트로 열려있는 서비스에 요청을 보내는 모습이다.
imgString이라는 데이터를 전송하고 결과물은 JsonVO.class형태로 받는다.



```
package com.multicampus.finalproject.model;

import java.util.ArrayList;
import java.util.List;

import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
@Getter
public class JsonVO {
    @XmlElement(name="response_img")
    private String response_img="default";
    @XmlElement(name="labels")
    private ArrayList<String> labels;
}
```

JsonVO의 형태이다. 이는 요청에 대한 응답으로 Json형태의 데이터에서
@XxmlElement annotation을 통해 Json의 key값과 변수를 1:1매핑해준다.

Controller를 작성하여 개별적으로 사용해보도록한다.

Made by 장경석