스프링 부트 웹 프로젝트

chapter11

Open API 사용하기

제공된 자료는 훈련생의 수업을 돕기 위한 것으로, 타인과 공유하시면 안됩니다.

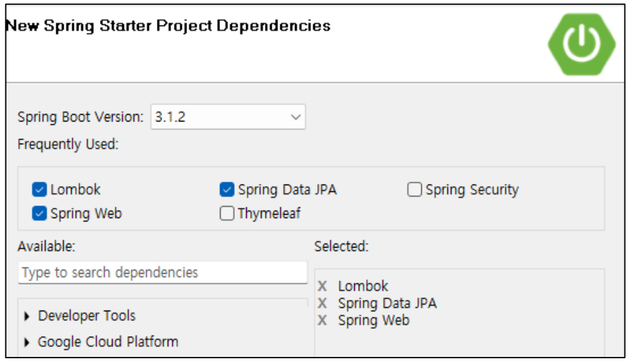
Contents

Open API

프로젝트 생성

11번째 프로젝트를 생성한다.

라이브러리는 'Lombok, Spring Web'를 선택한다.



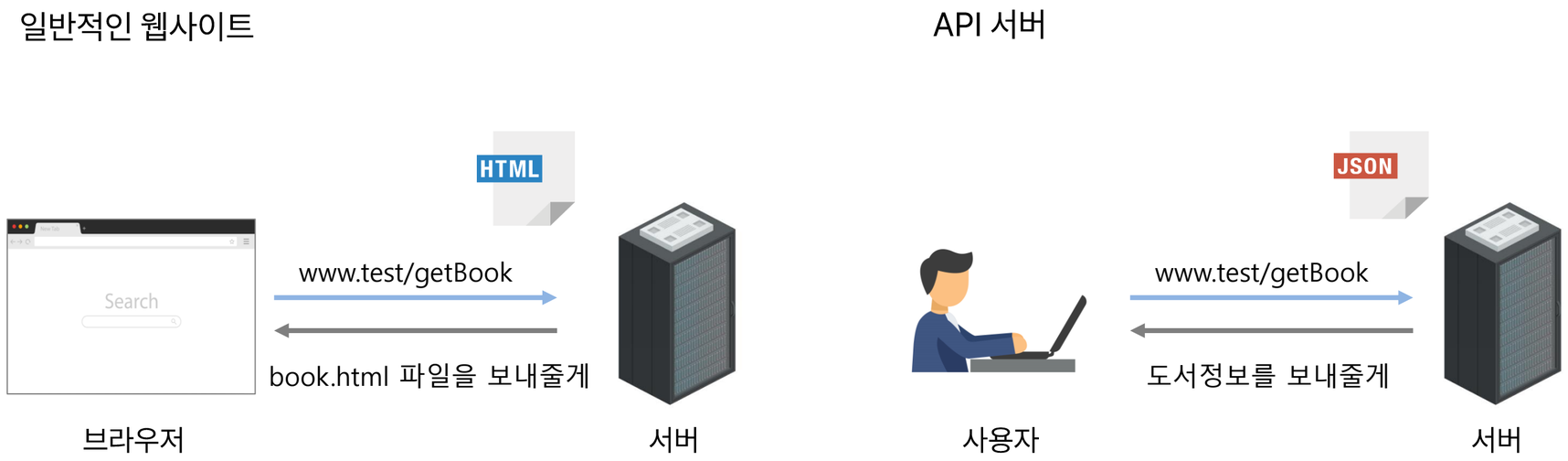
Open API

API란?

API

API는 다른 어플리케이션이나 사용자에게 데이터를 제공하는 어플리케이션이다.

일반 웹사이트는 브라우저를 통해 시각적으로 정보를 제공하지만, API는 데이터를 직접 전달한다.



Open API

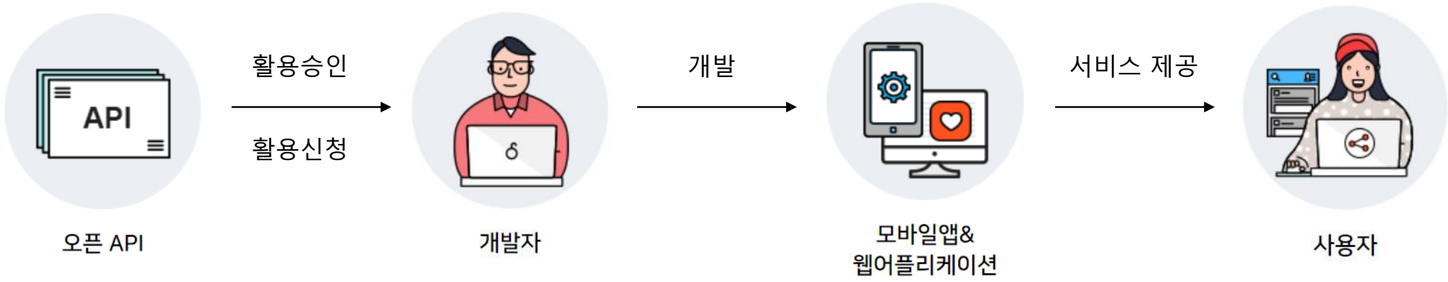
Open API란?

Open API

Open API는 누구나 사용할 수 있도록 공개된 API이다.

개발자들은 이를 활용하여 다양한 서비스를 개발할 수 있다.

공공기관은 데이터의 활용성을 높이기 위해 API를 통해 데이터를 제공한다.



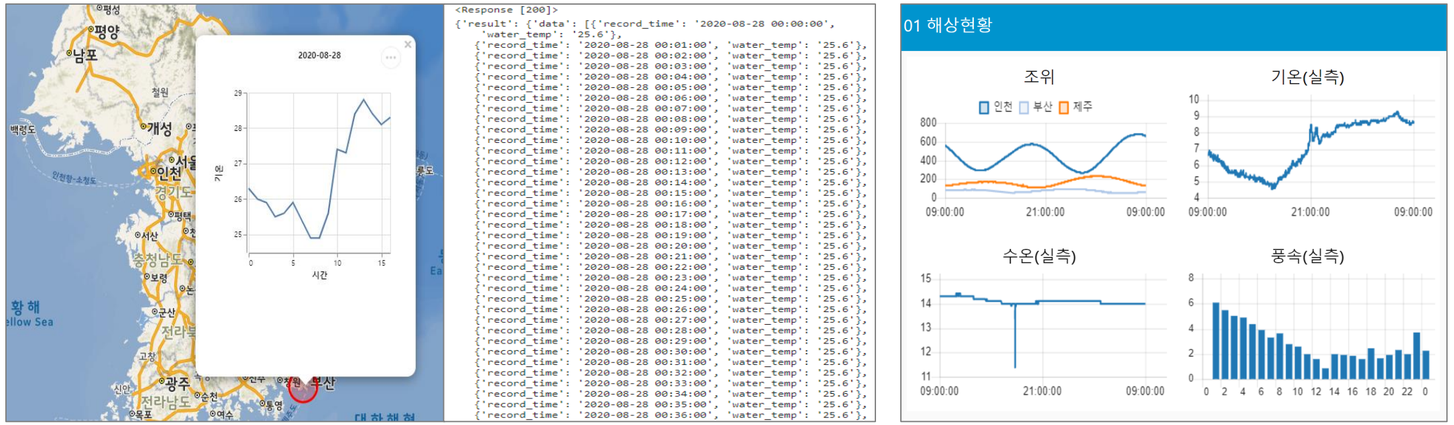
제공처: 공공데이터, Naver, Google 등

Open API

Open API란?

Open API를 활용한 서비스

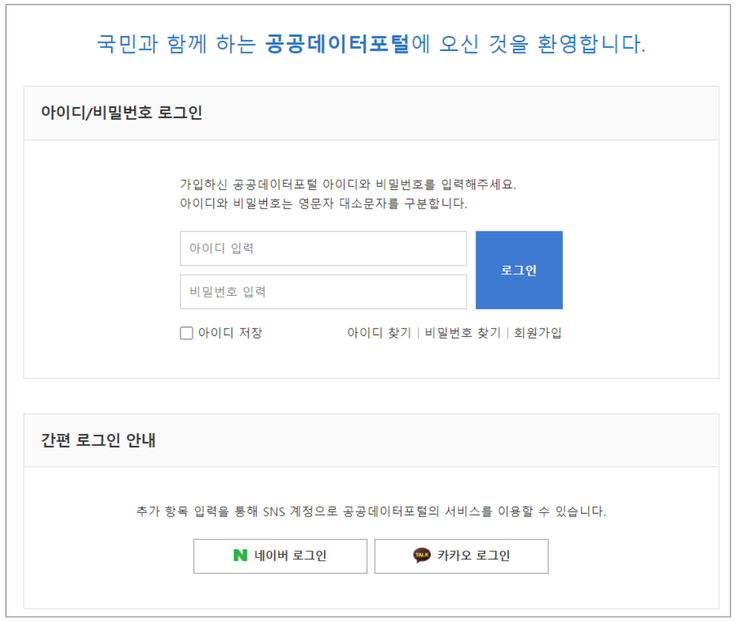
예를 들어 기상청이 제공하는 날씨 정보를 활용하여 기간별 통계 서비스를 만들 수 있다.



날씨정보를 활용한 통계 서비스

공공 데이터 포털 회원가입하기

공공데이터 포털에 접속해서 회원가입을 한다.



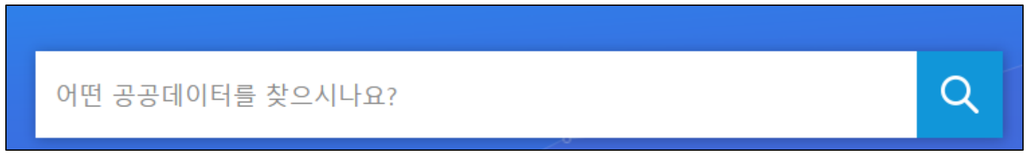
https://www.data.go.kr/

Open API 사용

API 조회

"기상청 동네예보 통보문"을 검색한다.

API를 선택한다.



Open API 사용 API 조회

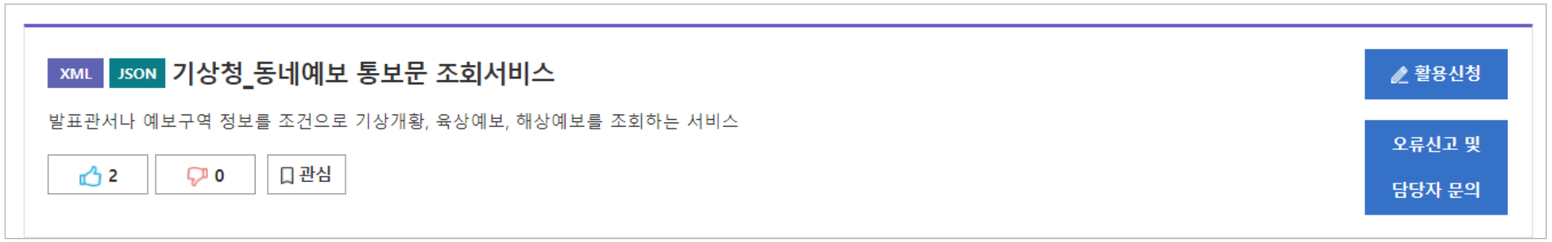
동네 예보 API 조회한다.
동네예보 API는 기상청에서 수집한 날씨 정보를 제공한다.
API의 제공기관, 데이터포맷, API 유형 등을 확인한다.
API에서 제공하는 데이터 종류를 확인한다.
육상예보조회 기능을 선택하고, 요청변수와 출력결과를 확인한다.

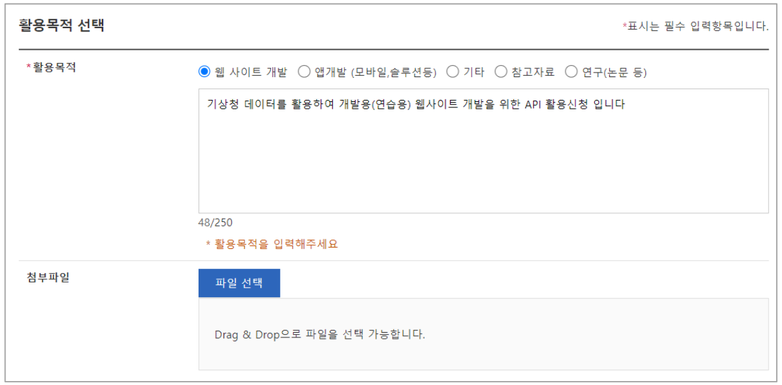
Open API 사용

API 신청하기

동네 예보 API 활용 신청한다.

1. [활용신청] 버튼을 클릭한다.
2. 활용목적은 '웹사이트개발'을 선택하고 내용은 간단하게 작성한다.
3. 해상예보조회, 육상예보조회, 기상개황조회 3개 기능을 모두 선택한다.
4. [활용신청] 버튼을 클릭하면 바로 신청이 승인된다.





Open API 사용

API 신청하기

5. 마이페이지 > 데이터 활용 > open api > 활용신청 현황 메뉴로 이동한다.
6. 신청결과에서 승인되었는지 확인한다.
7. 서비스를 선택한다.
8. 서비스 정보에서 인증키를 확인한다. 인증키는 API 호출시 사용된다.

Open API 사용하기

API 가이드 보기

API를 이용하려면 API를 호출하는 방법을 알아야 한다

가이드 문서에는 API에 대한 정보가 자세히 작성되어 있다.

- API주소, 기능별 상세주소, 필요한 파라미터, 응답 샘플 데이터

가이드 문서

서비스정보

참고문서	기상청19 동네예보 통보문 조회서비스 오픈API활용가이드.zip

내부적으로 사용하는 코드

개황	제주도	제주	184
육상	서울·인천·경기도	서울	11B10101
육상	서울·인천·경기도	인천	11B20201
육상	서울·인천·경기도	수원	11B20601
육상	서울·인천·경기도	성남	11B20605
육상	서울·인천·경기도	안양	11B20602
육상	서울·인천·경기도	광명	11B10103

Open API 사용

Open API 사용하기

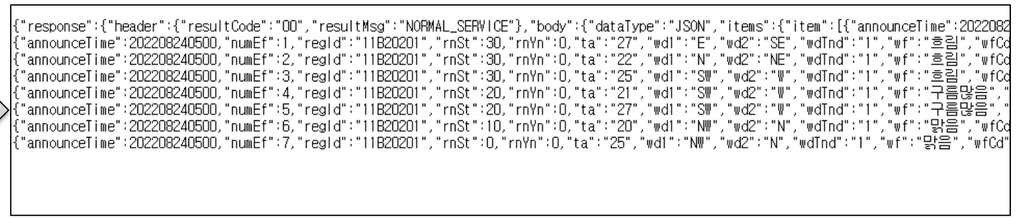
API 미리 사용해보기

1. 마이페이지 > 데이터 활용 > open api > 활용신청 현황 메뉴로 이동한다.
2. 서비스를 선택한다.
3. 활용 신청 상세기능 정보에서 육상예보 조회 – [확인] 버튼을 클릭한다.
4. 데이터포맷은 “JSON” 지역코드는 “11B20201”을 입력하고, [미리보기] 버튼을 클릭한다.
5. API를 호출하고 응답결과를 확인한다. 해당 데이터는 가장 최근에 발표된 육상예보이다.

2 육상예보조회 발표된 육상예보 중 가장 최근 예보드, 발표시각, 발효번호, 풍향, 풍향강도코드, 예상기온, 강수확률, 날씨수형태의 정보를 조회하는 기능

요청변수(Request Parameter)

항목명	샘플데이터	
ServiceKey	-	공공데이터포털에서 받은 인증키
pageNo	1	페이지번호
numOfRows	10	한 페이지 결과 수
dataType	JSON	요청자료형식(XML/JSON) Default: X
regId	11B20201	11A00101(백령도), 11B10101 (서울), 구분 값 참고)



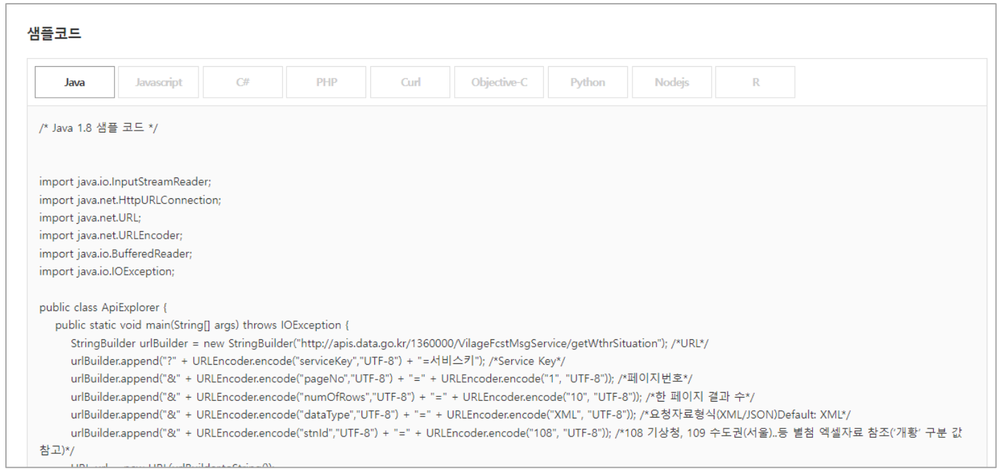
{"response":{"header":{"resultCode":"00","resultMsg":"NORMAL_SERVICE"},"body":{"dataType":"JSON","items":{"item":[{"announceTime":2022082
{"announceTime":202208240500,"numEf":1,"regId":"11B20201","rnSt":30,"rnYn":0,"ta":"27","wd1":"E","wd2":"SE","wdTnd":"1","wf":"흐림","wfCd
{"announceTime":202208240500,"numEf":2,"regId":"11B20201","rnSt":30,"rnYn":0,"ta":"22","wd1":"N","wd2":"NE","wdTnd":"1","wf":"흐림","wfCd
{"announceTime":202208240500,"numEf":3,"regId":"11B20201","rnSt":30,"rnYn":0,"ta":"25","wd1":"SW","wd2":"W","wdTnd":"1","wf":"흐림","wfCd
{"announceTime":202208240500,"numEf":4,"regId":"11B20201","rnSt":20,"rnYn":0,"ta":"21","wd1":"SW","wd2":"W","wdTnd":"1","wf":"구름많음","
{"announceTime":202208240500,"numEf":5,"regId":"11B20201","rnSt":20,"rnYn":0,"ta":"27","wd1":"SW","wd2":"W","wdTnd":"1","wf":"구름많음","
{"announceTime":202208240500,"numEf":6,"regId":"11B20201","rnSt":10,"rnYn":0,"ta":"20","wd1":"NW","wd2":"N","wdTnd":"1","wf":"맑음","wfCd
{"announceTime":202208240500,"numEf":7,"regId":"11B20201","rnSt":0,"rnYn":0,"ta":"25","wd1":"NW","wd2":"N","wdTnd":"1","wf":"맑음","wfCd"

Open API 사용

Open API 사용하기

자바 프로그램에서 API 호출하기

1. 다시 동네예보 서비스의 상세페이지로 이동한다.
2. 상세기능이 육상예보로 선택되어 있는지 확인 한다.
3. 페이지 하단에서 Java 샘플 코드를 찾아서 복사한다.
4. 샘플코드를 사용하여 자바코드로 API를 호출한다.



샘플코드

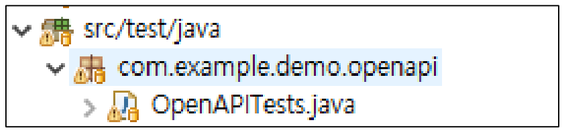
Java | Javascript | C# | PHP | Curl | Objective-C | Python | Nodejs | R

```
/* Java 1.8 샘플 코드 */


import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLEncoder;
import java.io.BufferedReader;
import java.io.IOException;

public class ApiExplorer {
    public static void main(String[] args) throws IOException {
        StringBuilder urlBuilder = new StringBuilder("http://apis.data.go.kr/1360000/VilageFcstMsgService/getWthrSituation"); /*URL*/
        urlBuilder.append("?" + URLEncoder.encode("serviceKey","UTF-8") + "=서비스키"); /*Service Key*/
        urlBuilder.append("&" + URLEncoder.encode("pageNo","UTF-8") + "=" + URLEncoder.encode("1", "UTF-8")); /*페이지번호*/
        urlBuilder.append("&" + URLEncoder.encode("numOfRows","UTF-8") + "=" + URLEncoder.encode("10", "UTF-8")); /*한 페이지 결과 수*/
        urlBuilder.append("&" + URLEncoder.encode("dataType","UTF-8") + "=" + URLEncoder.encode("XML", "UTF-8")); /*요청자료형식(XML/JSON)Default: XML*/
        urlBuilder.append("&" + URLEncoder.encode("stnId","UTF-8") + "=" + URLEncoder.encode("108", "UTF-8")); /*108 기상청, 109 수도권(서울)..등 별첨 엑셀자료 참조('개황' 구분 값
참고)*/
```

Open API 사용하기

openapi 패키지를 생성하고, 밑에 OpenAPITest 클래스를 생성한다.
클래스 상단에 API 호출에 사용할 파라미터를 선언한다.
인증키, 데이터타입, 지역코드 변수를 선언하고 값을 저장한다.

패키지



OpenAPITests

```
@SpringBootTest
public class OpenAPITests {
    String serviceKey = "EHoOALnbGNikccSKc4nAdZmHeHC…"
    String dataType = "JSON";
    String code = "11B20201";
```

Open API 사용하기

메소드 안에 API의 샘플 코드를 넣는다.

파라미터를 변수로 변경한다.

OpenAPITests

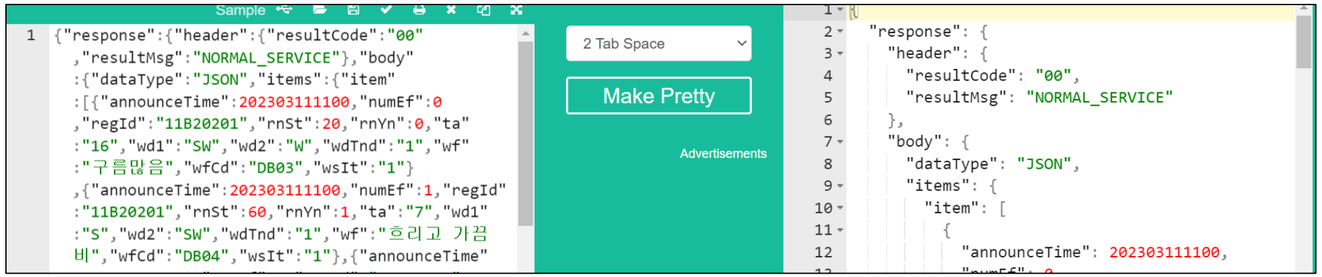
```
@Test
public void getWeather(){

}
```

```
urlBuilder.append("?" + URLEncoder.encode("serviceKey“)+ "="+serviceKey);
urlBuilder.append("&" + URLEncoder.encode("dataType") + "=" + URLEncoder.encode(dataType,));
urlBuilder.append("&" + URLEncoder.encode("regId") + "=" + URLEncoder.encode(code));
```

Open API 사용하기

테스트

- 단위 테스트를 실행한다.
- 200 응답코드와 일기예보 데이터를 응답 받았는지 확인한다.
- 콘솔창에서 일기예보 데이터를 복사한다.
- https://jsonformatter.org/json-pretty-print 사이트에 접속한다.
- 왼쪽칸에 복사한 JSON 데이터를 입력하면, 오른쪽에 정렬된 JSON 데이터가 출력된다.



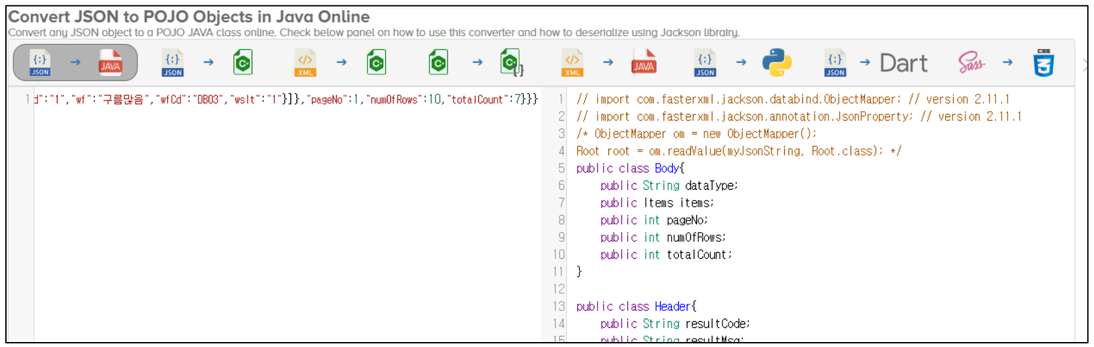
Open API 사용 JSON 데이터 파싱하기

JSON 문자열을 클래스로 변경하기

API로 받은 JSON 데이터는 바로 사용할 수 없으므로, 이를 파싱하여 Java 객체로 변환해야 한다.

1. https://json2csharp.com/code-converters/json-to-pojo 변환사이트에 접속한다.
2. 왼쪽 패널에 JSON문자열을 넣고, [convert] 버튼을 누른다.
3. 오른쪽 패널에 출력된 자바 클래스 코드를 복사한다.

```
Response code: 200
{"response":{"header":{"resultCode":"00","resultMsg":"NORMAL_SERVICE"},'
```

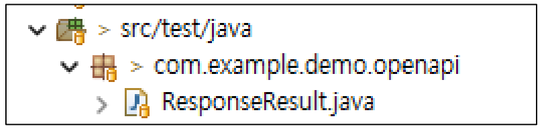


Open API 사용 JSON 데이터 파싱하기

JSON 문자열을 저장할 클래스 생성

1. openapi 패키지 밑에 ResponseResult 클래스를 생성한다.
2. 변환한 자바 코드를 붙여 넣는다.
   한 파일 안에 여러 클래스를 작성한다. 각 클래스는 JSON 응답의 구조가 정의되어 있다.

패키지



ResponseResult

```java
package com.example.demo.openapi;

import java.util.ArrayList;

@ToString
class Body{
    public String dataType;
    public Items items;
    public int pageNo;
    public int numOfRows;
    public int totalCount;
}

@ToString
class Header{
    public String resultCode;
    public String resultMsg;
}
```

public 접근제어자를 지워주세요!

```
{
  "response": {
    "header": {},
    "body": {}
  }
}
```

```
"body": {
  "dataType": "JSON",
  "items": {},
  "pageNo": 1,
  "numOfRows": 10,
  "totalCount": 7
}
```

```
"header": {
  "resultCode": "00",
  "resultMsg": "NORMAL_SERVICE"
},
```

```
"items": {
  "item": [
    {
      "announceTime": 202303111100,
      "numEf": 0,
      "regId": "11B20201",
      "rnSt": 20,
      "rnYn": 0,
      "ta": "16",
      "wd1": "SW",
      "wd2": "W",
      "wdTnd": "1"
```

Open API 사용 JSON 데이터 파싱하기

JSON 파싱하기

getWeather() 단위 테스트를 수정하여 리턴타입을 문자열로 변경하고, API 결과를 반환한다.

JSON문자열을 Java 클래스로 변환하는 메소드를 추가한다.

Jackson 라이브러리의 ObjectMapper 클래스를 사용하여 JSON문자열을 Java 객체로 변환한다.

OpenAPITest

```
public String getWeather() throws IOException {
    …
    return sb.toString();
}
```

```
@Test
public void jsonToDto() throws IOException {
      …
    String weather = getWeather();
    ResponseResult response = null;
    response = mapper.readValue(weather, ResponseResult.class);
}
```