



Open API 샘플 가이드

작성일: 2022.04.06

버전: v1.0

목록

1장. Open API 기본 설명

2장. Java 1.8 샘플 코드

3장. JavaScript 샘플 코드

4장. cURL 샘플 코드


5장. Python 3 샘플 코드

6장. Node.js 12 샘플 코드

※ 모든 예시는 서울시 지하철호선별 역별 승하차 인원 정보를 기준으로 사용/활용하고 있습니다.

※ 테스트 OS 환경은 **Windows 10** 입니다.


1장. Open API 기본 설명


서울 열린데이터 광장
공공데이터
통계
소식&참여
나의화면
이용안내
AI학습데이터
로그아웃
사이트맵

인증키 신청

Home > 나의화면 > 인증키 신청

인증키 안내



열린데이터광장에서 제공하는 오픈API를 사용하기 위해서는 먼저 인증키를 발급받으셔야 합니다.
오픈API는 다양한 서비스와 데이터를 좀 더 쉽게 이용할 수 있도록 공개한 개발자를 위한 인터페이스입니다.

◆ 인증키 발급

- 열린데이터광장에서 제공하는 오픈API를 사용하기 위해서는 먼저 인증키를 발급 받으셔야 합니다.
- 오픈API는 다양한 서비스와 데이터를 좀 더 쉽게 이용할 수 있도록 공개한 개발자를 위한 인터페이스입니다.

◆ 인증키 사용

- 실시간 지하철 오픈API의 경우 하루에 최대 1,000회 요청할 수 있습니다.
- 오픈API를 통해서 1회에 최대 1,000건을 요청할 수 있습니다.
(1,000건이 넘는 경우 나누어서 호출하세요.)
- 이용 제약 없이 계속 사용하시려면 서울데이터 활용갤러리에 인증키와 함께 콘텐츠를 등록하세요.

일반 인증키 신청

실시간 지하철 인증키 신청

서울시 열린데이터광장의 인증키는 **일반 인증키**와 **실시간 지하철 인증키** 두 가지로 구분되어 있습니다.

구분	일반 인증키	실시간 지하철 인증키
도메인	http://openapi.seoul.go.kr	http://swopenapi.seoul.go.kr
특이점	<ul style="list-style-type: none"> 서비스 별로 신청할 필요 없음 1회 호출 시 최대 1,000건 가능 	<ul style="list-style-type: none"> 하루 최대 1,000회 호출 가능 활용갤러리 등록 시 1,000회 제한 해제 1회 호출 시 최대 1,000건 가능

인증키를 사용하기 위해서는 로그인が必要です. 서울시 홈페이지 회원이신 분은 동일한 아이디와 비밀번호로 로그인 후 이용 가능하며, 회원이 아니신 경우 서울시 홈페이지에서 가입 및 로그인 후 이용 가능합니다.

[서울시 홈페이지 회원가입 바로가기 >](#)

Sheet

Open API

1

> 샘플 URL

Open API 이용안내

인증키 신청

명세서다운로드

샘플 URL 2015년11월1일 지하철호선별 역별 승하차 인원
http://openapi.seoul.go.kr:8088/{인증키}/xml/CardSubwayStatsNew/1/5/20151101

예제

```
<?xml version="1.0" encoding="UTF-8"?>
<CardSubwayStatsNew>
<list_total_count>548</list_total_count>
<RESULT>
<CODE>INFO-000</CODE>
<MESSAGE>정상 처리되었습니다</MESSAGE>
</RESULT>
<row>
<USE_DT>20151101</USE_DT>
<LINE_NUM>1호선</LINE_NUM>
<SUB_STA_NM>1호선</SUB_STA_NM>
```

2

> 요청인자

	변수명	타입	변수설명	값설명
A	KEY	String(필수)	인증키	OpenAPI 에서 발급된 인증키
B	TYPE	String(필수)	요청파일타입	xml:xml,xml파일:xmlf, 엑셀파일:xls,json파일:json
C	SERVICE	String(필수)	서비스명	CardSubwayStatsNew
D	START_INDEX	INTEGER(필수)	요청시작위치	정수 입력 (페이징 시작번호 입니다:데이터 행 시작번호)
E	END_INDEX	INTEGER(필수)	요청종료위치	정수 입력 (페이징 끝번호 입니다:데이터 행 끝번호)
F	USE_DT	STRING(필수)	사용일자	YYYYMMDD 형식의 문자열

3

> 출력값

No	출력명	출력설명
공통	list_total_count	총 데이터 건수 (정상조회시 출력됨)
공통	RESULT.CODE	요청결과 코드 (하단 예제자설명 참고)
공통	RESULT.MESSAGE	요청결과 메시지 (하단 예제자설명 참고)
1	USE_DT	사용일자
2	LINE_NUM	호선명
4	SUB_STA_NM	역명
5	RIDE_PASGR_NUM	승차총승객수
6	ALIGHT_PASGR_NUM	하차총승객수
7	WORK_DT	등록일자

[예제 페이지 바로가기 >](#)

화면설명

- 1. 샘플 URL: Open API 샘플코드가 예제에 정상적으로 나오는지 확인할 수 있습니다.
- 2. 요청인자: Open API 호출 시 필요한 인자를 확인할 수 있습니다.
특히 타입에 (필수) 부분은 필수로 호출해야 하므로 잊지 않아야 합니다. 순서는 꼭 지켜서 호출해야 합니다.
- 2.1. KEY: 발급 받은 인증키이며 sample 사용 시 최대 5건으로 제한됩니다.
- 2.2. TYPE: 출력되는 타입을 표시합니다.
(xml: xml, xml파일: xmlf, 엑셀파일: xls, json파일: json)
- 2.3. SERVICE: 각 서비스 별 고정된 값
- 2.4. START_INDEX: 데이터 행 시작번호입니다.
(sample 키 사용 시 1~5로 제한)
- 2.5. END_INDEX: 데이터 행 끝번호입니다.
(sample 키 사용 시 1~5로 제한)
END_INDEX - START_INDEX가 999를 넘을 수 없습니다.

※ 아래 인자부터는 인자의 유무가 서비스에 따라 상이합니다.

- 2.6. USE_DT: 타입에서 (선택), (필수)로 구분되어 있으며, (선택)일 경우 입력하지 않아도 API는 정상적으로 호출됩니다.
단, 선택인자의 일부를 사용하는 경우 순서를 지켜서 사용해 주시기 바랍니다.

★ 선택인자 호출 예시

예) 자치구단위 서울 생활인구(내국인)

> 요청인자

변수명	타입	변수설명	값설명
KEY	String(필수)	인증키	OpenAPI에서발급된인증키
TYPE	String(필수)	요청파일타입	xml:xml,xml파일:xmlf,엑셀파일:xls,json파일:json
SERVICE	String(필수)	서비스명	SPOP_LOCAL_RESD_JACHI
START_INDEX	INTEGER(필수)	요청시작위치	정수입력(페이징 시작번호 입니다:데이터 행 시작번호)
END_INDEX	INTEGER(필수)	요청종료위치	정수입력(페이징 끝번호 입니다:데이터 행 끝번호)
STDR_DE_ID	STRING(선택)	기준일ID	
TMZON_PD_SE	STRING(선택)	시간대구분	
ADSTRD_CODE_SE	STRING(선택)	자치구코드	

자치구코드 부분 사용 시, 기준일ID, 시간대구분 부분은 공백으로 호출해야 합니다.

http://openapi.seoul.go.kr:8088/sample/xml/SPOP_LOCAL_RESD_JACHI/1/5//11110
http://openapi.seoul.go.kr:8088/sample/xml/SPOP_LOCAL_RESD_JACHI/1/5/

3. 출력값: Open API 호출 후 나오는 출력값을 확인할 수 있습니다.

<?xml version="1.0" encoding="UTF-8"?>	<== xml 인코딩 선언입니다.
<CardSubwayStatsNew>	<== 서비스명 입니다.
<list_total_count>548</list_total_count>	<== '총 데이터 건수'로써 이를 바탕으로 1000건씩 분리해서 호출할지 판단합니다.
<RESULT>	
<CODE>INFO-000</CODE>	<== '요청결과 코드'로써 ERROR 발생시에는 열광서비스 오류입니다.
<MESSAGE>정상 처리되었습니다</MESSAGE>	<== '요청결과 메시지'로써 '요청결과 코드'에 따른 메시지입니다.
</RESULT>	
<row>	

List_total_count가 1,000이 넘을 경우 Open API는 1회에 1,000건을 넘을 수 없으므로 분리해서 호출합니다.

2장. Java 1.8 샘플 코드

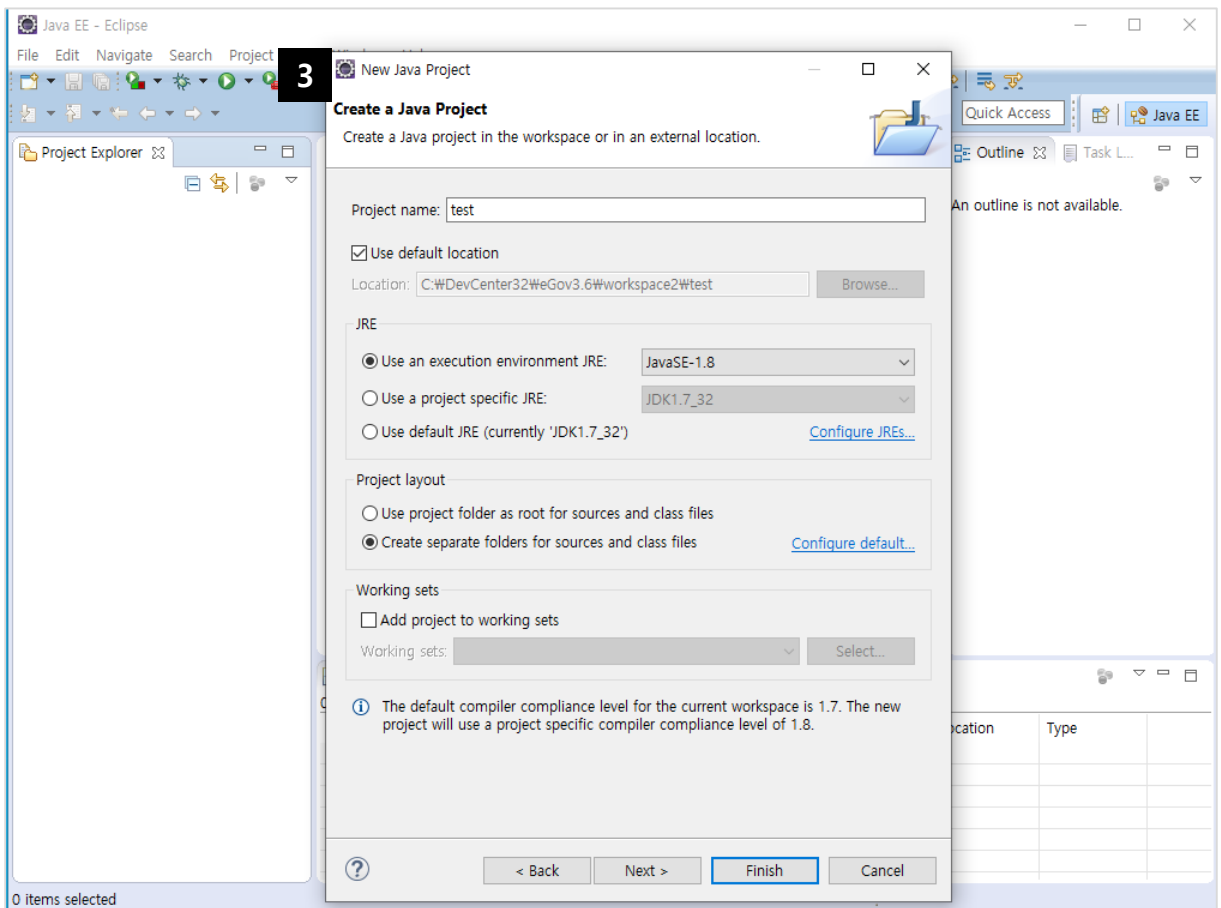
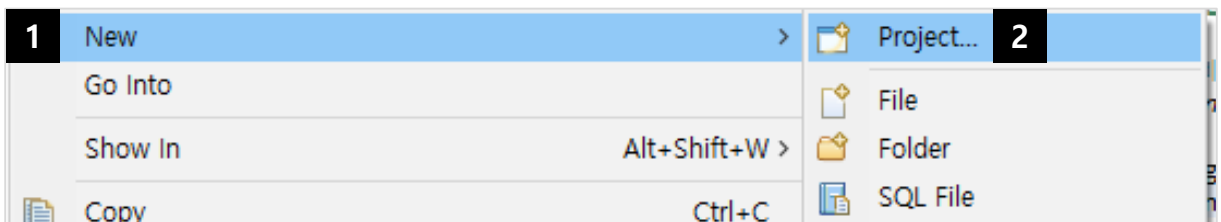
1. 이클립스 다운로드 및 실행

이클립스 홈페이지에서 이클립스를 다운로드 및 설치한 후 eclipse.exe를 실행합니다.

[이클립스 홈페이지 바로가기 >](#)

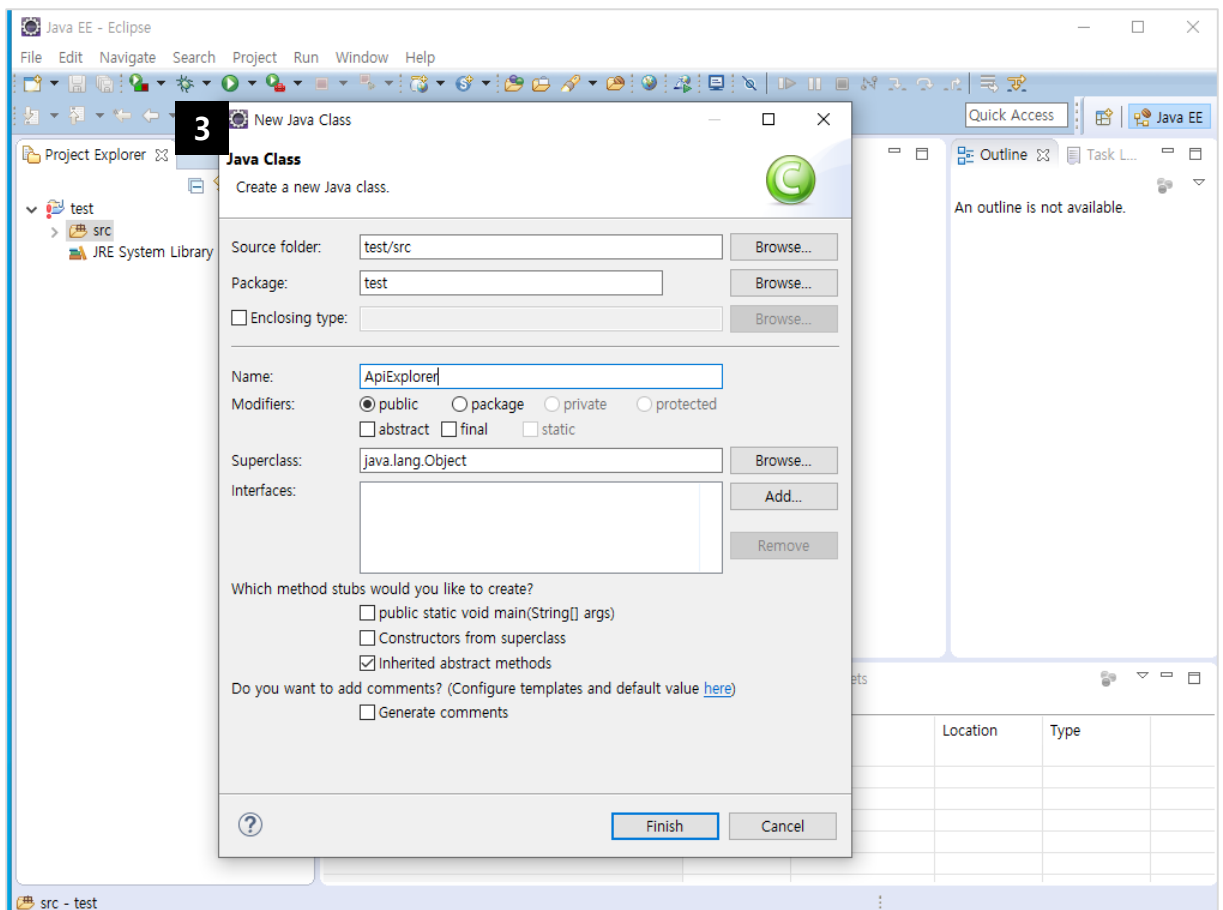
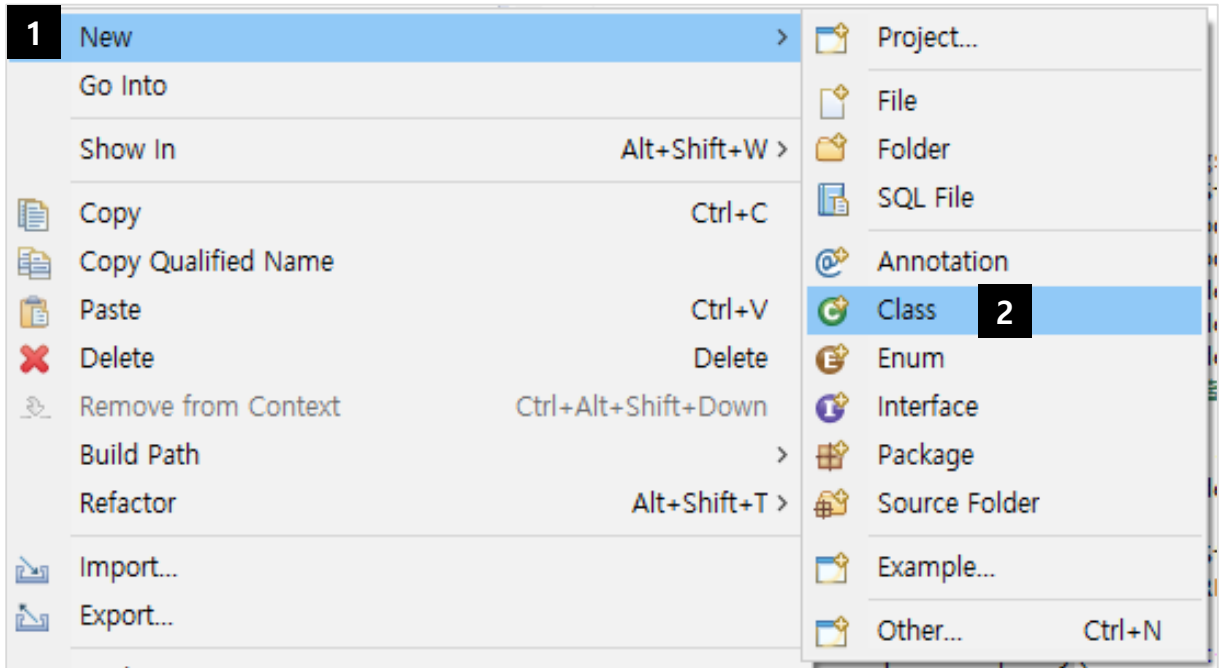
2. Java Project 생성하기

빈 Project Explorer 영역에서 마우스 우클릭 후 ① **New** → ② **Project**를 클릭하여 ③ 프로젝트를 생성합니다.



3. Java Class 생성하기

Project Explorer 영역에서 생성한 프로젝트에 우클릭 후 ① **New** → ② **Class**를 클릭하여 ③ **클래스를 생성**합니다.



4. 샘플 코드

아래의 샘플 코드를 복사하여 생성한 Class의 편집창에 붙여 넣습니다.

```
package test;

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://openapi.seoul.go.kr:8088");

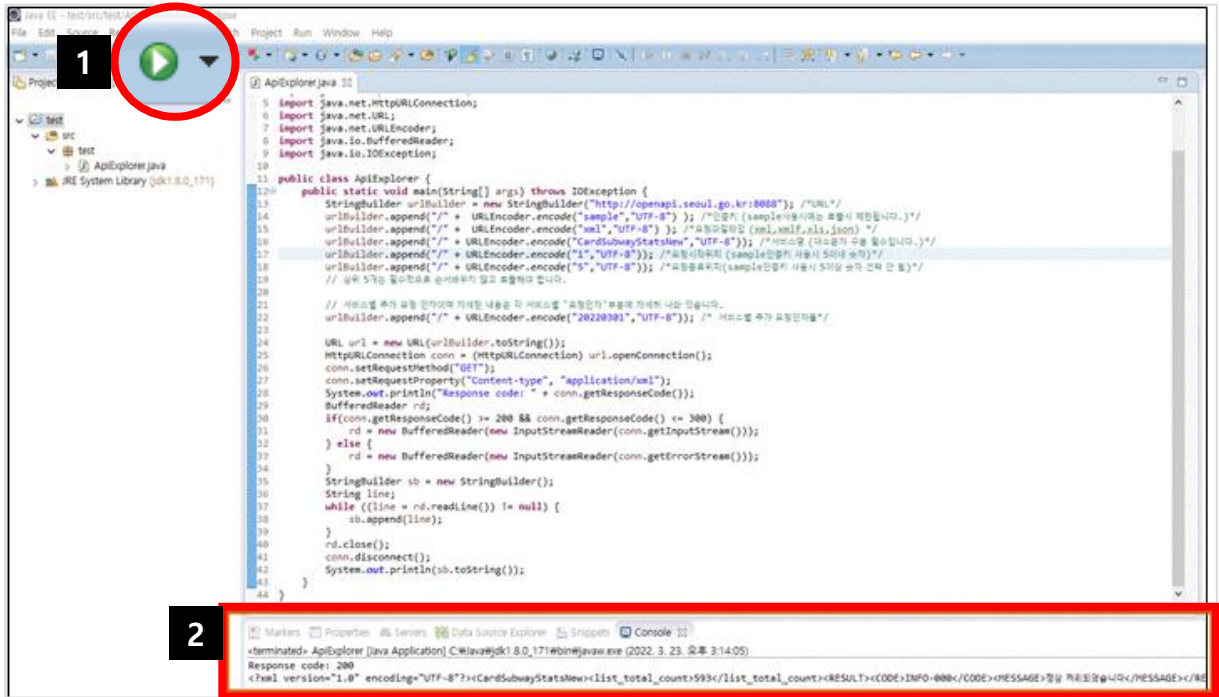
        /*URL*/
        urlBuilder.append("/") + URLEncoder.encode("sample","UTF-8") ); /*인증키
(sample사용시에는 호출시 제한됩니다.)*/
        urlBuilder.append("/") + URLEncoder.encode("xml","UTF-8") ); /*요청파일타입
(xml,xm1f,xls,json) */
        urlBuilder.append("/") + URLEncoder.encode("CardSubwayStatsNew","UTF-8"));
        /*서비스명 (대소문자 구분 필수입니다.)*/
        urlBuilder.append("/") + URLEncoder.encode("1","UTF-8")); /*요청시작위치
(sample인증키 사용시 5이내 숫자)*/
        urlBuilder.append("/") + URLEncoder.encode("5","UTF-8"));
        /*요청종료위치(sample인증키 사용시 5이상 숫자 선택 안 됨)*/
        // 상위 5개는 필수적으로 순서바꾸지 않고 호출해야 합니다.
        // 서비스별 추가 요청 인자이며 자세한 내용은 각 서비스별 '요청인자'부분에
        자세히 나와 있습니다.
        urlBuilder.append("/") + URLEncoder.encode("20220301","UTF-8")); /* 서비스별
추가 요청인자들*/

        URL url = new URL(urlBuilder.toString());
        HttpURLConnection conn = (HttpURLConnection) url.openConnection();
        conn.setRequestMethod("GET");
        conn.setRequestProperty("Content-type", "application/xml");
        System.out.println("Response code: " + conn.getResponseCode()); /* 연결
자체에 대한 확인이 필요하므로 추가합니다.*/
        BufferedReader rd;

        // 서비스코드가 정상이면 200~300사이의 숫자가 나옵니다.
        if(conn.getResponseCode() >= 200 && conn.getResponseCode() <= 300) {
            rd = new BufferedReader(new InputStreamReader(conn.getInputStream()));
        } else {
            rd = new BufferedReader(new InputStreamReader(conn.getErrorStream()));
        }
        StringBuilder sb = new StringBuilder();
        String line;
        while ((line = rd.readLine()) != null) {
            sb.append(line);
        }
        rd.close();
        conn.disconnect();
        System.out.println(sb.toString());
    }
}
```

5. 실행 및 확인

① 실행 아이콘을 클릭 후 하단 ② Console창을 확인하여 결과가 정상적으로 나왔는지 확인합니다.



3장. JavaScript 샘플 코드

1. HTML 생성

원하는 위치에 메모장이나 워드패드 또는 별도의 HTML 편집기를 사용하여 확장자가 HTML인 파일을 생성합니다.

2. 샘플 코드

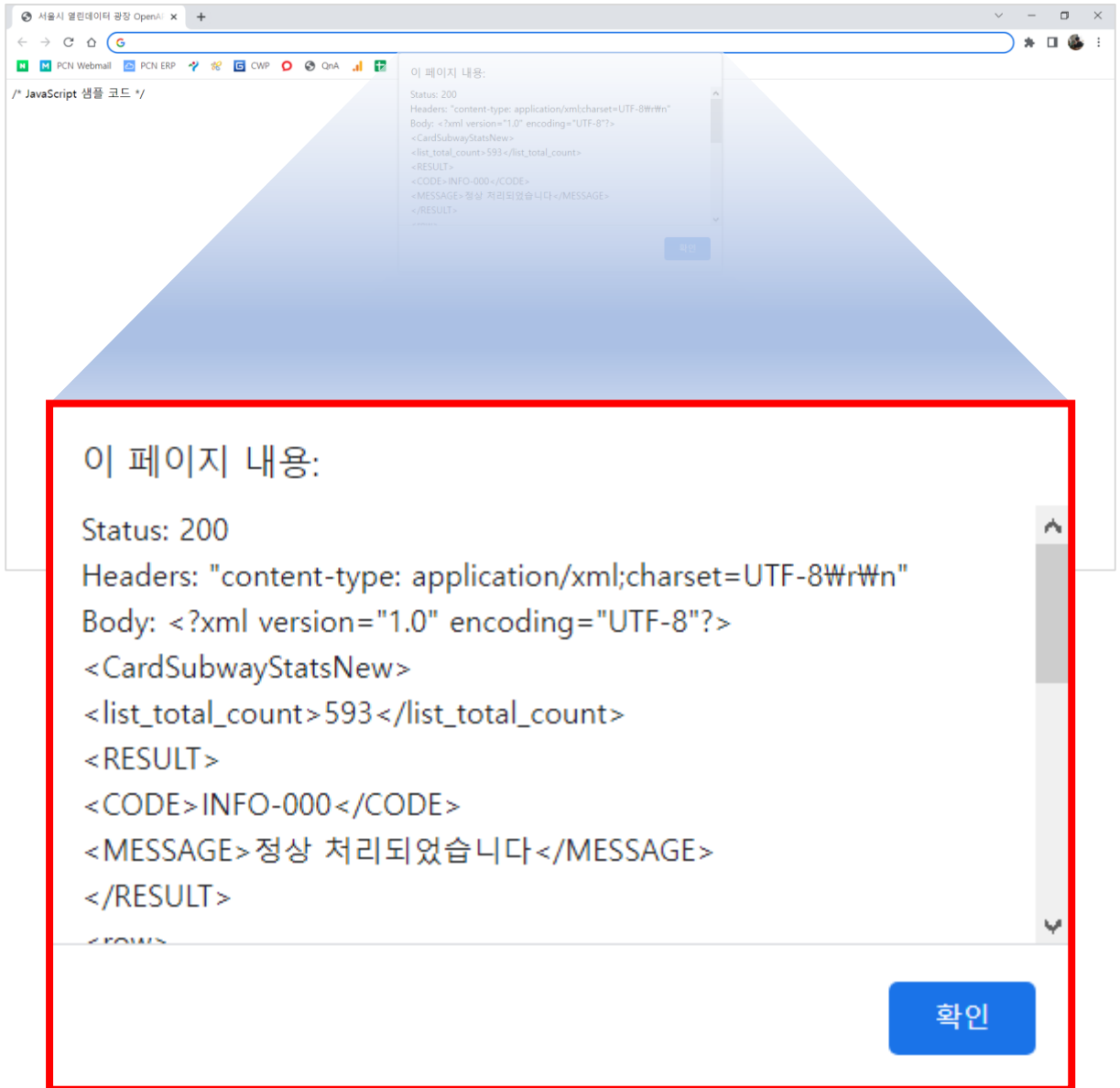
생성한 HTML 파일을 HTML 편집기(메모장, 워드패드 등)로 열고 아래의 샘플코드를 복사하여 붙여 넣습니다.

```
/* JavaScript 샘플 코드 */

<!DOCTYPE html>
<html lang="kr">
<head>
<meta charset="UTF-8">
<title>서울시 열린데이터 광장 OpenAPI 샘플(Javascript)</title>
</head>
<body>
<script>
var xhr = new XMLHttpRequest();
var url = 'http://openapi.seoul.go.kr:8088/sample/xml/CardSubwayStatsNew/1/5/20220301'; /*URL*/
xhr.open('GET', url);
xhr.onreadystatechange = function () {
if (this.readyState == xhr.DONE) { // <== 정상적으로 준비되었을때
    if(xhr.status == 200 | xhr.status == 201){ // <== 호출 상태가 정상적일때
        alert('Status: '+this.status+
            '\nHeaders: '+JSON.stringify(this.getAllResponseHeaders())+
            '\nBody: '+this.responseText);
    }
}
};
xhr.send("");
</script>
</body>
</html>
```

3. 확인

샘플 코드를 붙여넣은 HTML 파일을 저장하고 브라우저로 열어 API가 정상적으로 호출되는지 확인합니다.



The image shows a web browser window with a tab titled '서울시 열린데이터 광장 OpenAPI'. The address bar shows a Google search bar. Below the browser window, there is a large blue triangular graphic. In the foreground, a red-bordered box contains the following text:

이 페이지 내용:

Status: 200
Headers: "content-type: application/xml;charset=UTF-8WrWn"
Body: <?xml version="1.0" encoding="UTF-8"?>
<CardSubwayStatsNew>
<list_total_count>593</list_total_count>
<RESULT>
<CODE>INFO-000</CODE>
<MESSAGE>정상 처리되었습니다</MESSAGE>
</RESULT>
</FORM>

At the bottom right of the red-bordered box, there is a blue button with the text '확인' (Check).

4장. cURL 샘플 코드

1. cURL 다운로드

cURL 홈페이지에서 cURL을 다운로드하고 원하시는 위치에 압축을 풀어 놓습니다.
(예제에서는 C:\wcurl 경로에 압축을 풀었습니다.)



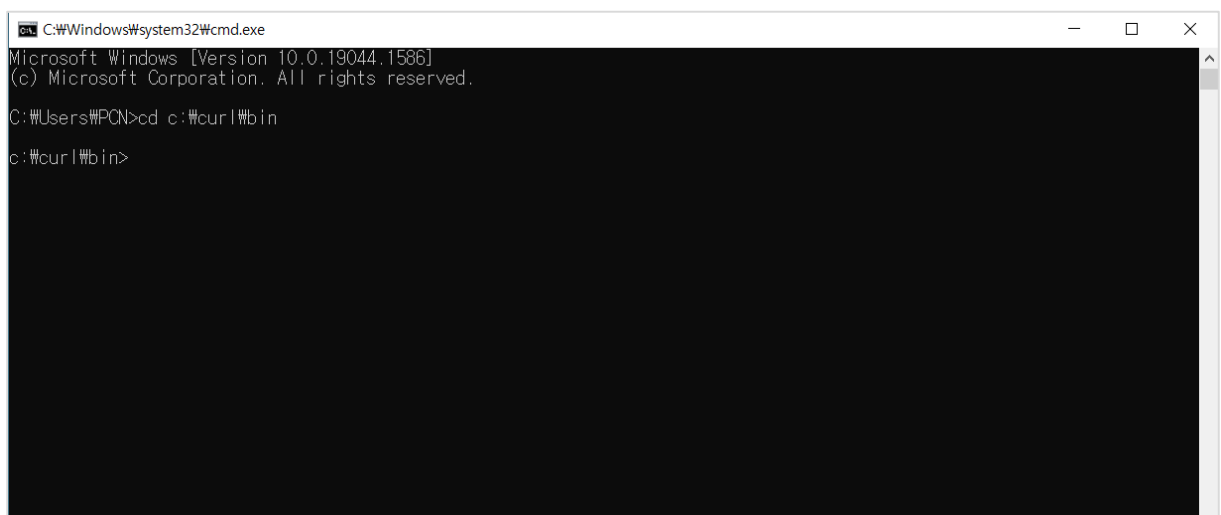
이름	수정된 날짜	유형	크기
bin	2022-03-30 오후 5:03	파일 폴더	
docs	2022-03-30 오후 5:03	파일 폴더	
include	2022-03-30 오후 5:03	파일 폴더	
lib	2022-03-30 오후 5:03	파일 폴더	
BUILD-HOMEPAGE	2022-03-05 오전 8:44	인터넷 바로 가기	1KB
BUILD-README.txt	2022-03-05 오전 8:44	텍스트 문서	1KB
CHANGES.txt	2022-03-05 오전 8:44	텍스트 문서	202KB
COPYING.txt	2022-03-01 오후 5:06	텍스트 문서	2KB
COPYING-brotli.txt	2020-08-27 오후 2:12	텍스트 문서	2KB
COPYING-libidn2.txt	2021-05-14 오후 7:54	텍스트 문서	2KB
COPYING-libssh2.txt	2021-08-29 오후 8:36	텍스트 문서	2KB
COPYING-nghttp2.txt	2022-02-23 오전 8:10	텍스트 문서	2KB
COPYING-openssl.txt	2022-03-15 오후 2:30	텍스트 문서	10KB
COPYING-zlib.txt	2022-03-27 오후 11:39	텍스트 문서	6KB
mk-ca-bundle.pl	2022-01-26 오전 9:09	PL 파일	21KB
README.txt	2022-01-26 오전 9:09	텍스트 문서	2KB
RELEASE-NOTES.txt	2022-03-05 오전 8:43	텍스트 문서	18KB

2. cURL 실행

윈도우키 + R키를 눌러 실행창을 띄운 뒤, cmd를 입력하고 엔터키를 눌러 명령 프롬프트를 실행합니다.

명령 프롬프트창이 뜨면 cURL의 bin 경로까지 이동합니다.

예제에서는 C:\wcurl\bin에 위치하므로 **cd c:\wcurl\bin**을 입력하여 이동합니다.



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19044.1586]
(c) Microsoft Corporation. All rights reserved.

C:\Users\PCN>cd c:\wcurl\bin
c:\wcurl\bin>
```

3. 샘플 코드 입력 및 확인

아래의 샘플 코드를 입력 및 실행하여 정상적으로 호출되는지 확인합니다.

```
curl --include --request GET
'http://openapi.seoul.go.kr:8088/sample/xml/CardSubwayStatsNew/1/5/20220301'
```

```
C:\#curl\#bin>curl --include --request GET http://openapi.seoul.go.kr:8088/sample/xml/CardSubwayStatsNew/1/5/20220301
HTTP/1.1 200 OK
Date: Wed, 30 Mar 2022 08:15:18 GMT
Server: Apache/2.2.26 (Red Hat Enterprise Web Server)
Set-Cookie: WMONID=Gsy5gZMHwKA; Expires=Thu, 30-Mar-2023 17:15:18 GMT; Path=/
Accept-Ranges: bytes
Access-Control-Allow-Origin: *
Connection: close
Transfer-Encoding: chunked
Content-Type: application/xml;charset=UTF-8

<?xml version="1.0" encoding="UTF-8"?>
<CardSubwayStatsNew>
<list_total_count>593</list_total_count>
<RESULT>
<CODE>INFO-000</CODE>
<MESSAGE>정상 처리되었습니다</MESSAGE>
</RESULT>
<row>
<USE_DT>20220301</USE_DT>
<LINE_NUM>1호선</LINE_NUM>
<SUB_STA_NM>서울역</SUB_STA_NM>
<RIDE_PASGR_NUM>20994</RIDE_PASGR_NUM>
<ALIGHT_PASGR_NUM>19468</ALIGHT_PASGR_NUM>
<WORK_DT>20220304</WORK_DT>
</row>
```

5장. Python 3 샘플 코드

1. Python 3 다운로드 및 설치

파이썬 홈페이지에서 파이썬을 다운로드하고 설치합니다.

[파이썬 홈페이지 바로가기 >](#)

2. Python 3에 requests를 설치

윈도우키 + R키를 눌러 실행창을 띄운 뒤, cmd를 입력하고 엔터키를 눌러 명령 프롬프트를 실행합니다.

아래와 같이 입력하여 request를 설치하기 위한 경로로 이동합니다.

cd c:\Users\<유저명>\AppData\Local\Programs\Python\Python38-32\Scripts

C:\Users\<유저명>\AppData\Local\Programs\Python\Python38-32\Scripts로 경로 이동이 완료되면 아래의 명령어를 입력하여 requests를 설치합니다.

pip install requests

```
C:\Users\<유저명>\AppData\Local\Programs\Python\Python38-32\Scripts>pip install requests
Collecting requests
  Downloading https://files.pythonhosted.org/packages/2d/61/08076519c80041bc0ffa1a8af0cd3bf3e2b62af10435cd63a8d0f40564d/requests-2.27.1-py2.py3-none-any.whl (63kB)
    |#####| 71kB 4.5MB/s
Collecting idna<4.0,=>2.5; python_version >= "3" (from requests)
  Downloading https://files.pythonhosted.org/packages/04/e2/d818dcd22354d8958fe113e1a3630137e0fc8b44859ade3063992eac02a4/idna-3.3-py3-none-any.whl (61kB)
    |#####| 61kB 3.0MB/s
Collecting urllib3<1.27,=>1.21.1 (from requests)
  Downloading https://files.pythonhosted.org/packages/ec/03/062e6444ce4ef1eac17a5a0ebf356b1ad05e1df0a20b110d58c278438/urllib3-1.26.9-py2.py3-none-any.whl (138kB)
    |#####| 143kB 6.4MB/s
Collecting charset-normalizer<2.0.0, python_version >= "3" (from requests)
  Downloading https://files.pythonhosted.org/packages/06/b3/24afcb888ebad69a7f08650ac750a778862dc34941a4bebe58706715725/charset-normalizer-2.0.12-py3-none-any.whl
Collecting certifi<2017.4.17 (from requests)
  Downloading https://files.pythonhosted.org/packages/37/45/345cd2767aebb673145011e665728b680884cd8fe70dd9b73cd540e45b775/certifi-2021.10.8-py2.py3-none-any.whl (143kB)
    |#####| 153kB 6.4MB/s
Installing collected packages: idna, urllib3, charset-normalizer, certifi, requests
WARNING: The script normalizer.exe is installed in 'c:\Users\<유저명>\AppData\Local\Programs\Python\Python38-32\Scripts' which is not on PATH.
Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed certifi-2021.10.8 charset-normalizer-2.0.12 idna-3.3 requests-2.27.1 urllib3-1.26.9
WARNING: You are using pip version 19.2.3, however version 22.0.4 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
C:\Users\<유저명>\AppData\Local\Programs\Python\Python38-32\Scripts>
```

3. 샘플 코드 입력 및 확인

Python을 실행하고 console에 아래의 샘플 코드를 입력 및 실행하여 정상적으로 호출되는지 확인합니다.

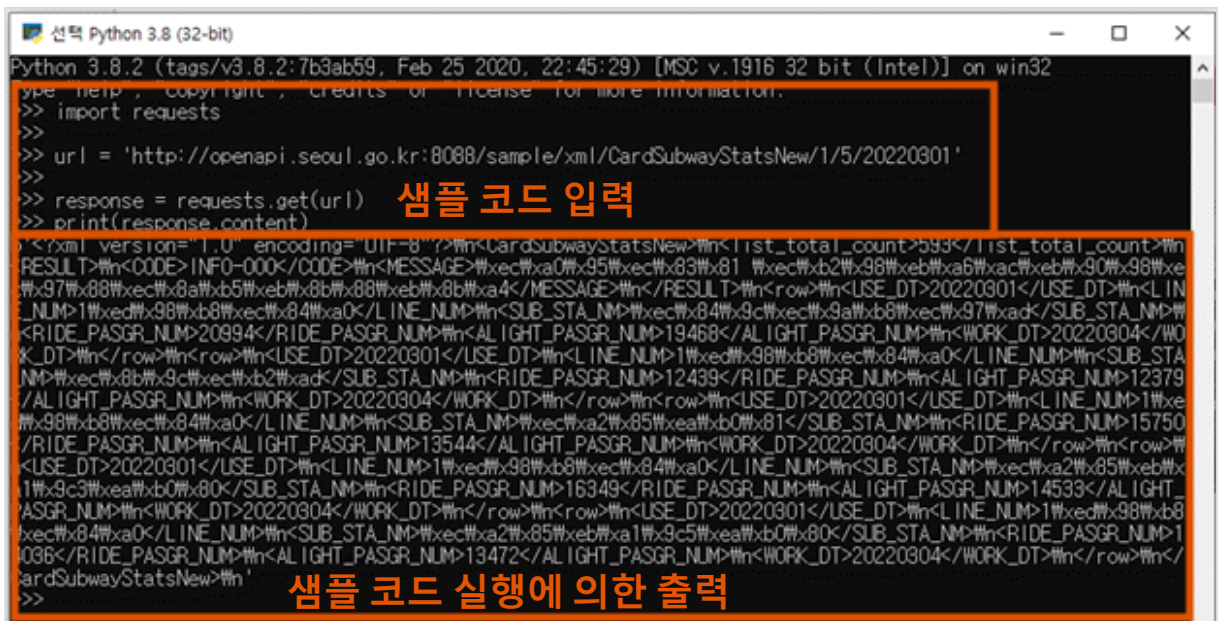
```
/* Python3 샘플 코드 */
```

```
import requests
```

```
url = 'http://openapi.seoul.go.kr:8088/sample/xml/CardSubwayStatsNew/1/5/20220301'
```

```
response = requests.get(url)
```

```
print(response.content)
```



```
선택 Python 3.8 (32-bit)
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
>>> import requests
>>> url = 'http://openapi.seoul.go.kr:8088/sample/xml/CardSubwayStatsNew/1/5/20220301'
>>> response = requests.get(url)
>>> print(response.content)
<?xml version="1.0" encoding="UTF-8"><CardSubwayStatsNew><list_total_count>593</list_total_count><RESULT><CODE>INFO-000</CODE><MESSAGE></MESSAGE></RESULT><row><USE_DT>20220301</USE_DT><LINE_NUM>1</LINE_NUM><SUB_STA_NM>1</SUB_STA_NM><RIDE_PASGR_NUM>20994</RIDE_PASGR_NUM><ALIGHT_PASGR_NUM>19468</ALIGHT_PASGR_NUM><WORK_DT>20220304</WORK_DT></row><row><USE_DT>20220301</USE_DT><LINE_NUM>1</LINE_NUM><SUB_STA_NM>1</SUB_STA_NM><RIDE_PASGR_NUM>12439</RIDE_PASGR_NUM><ALIGHT_PASGR_NUM>12379</ALIGHT_PASGR_NUM><WORK_DT>20220304</WORK_DT></row><row><USE_DT>20220301</USE_DT><LINE_NUM>1</LINE_NUM><SUB_STA_NM>1</SUB_STA_NM><RIDE_PASGR_NUM>15750</RIDE_PASGR_NUM><ALIGHT_PASGR_NUM>13544</ALIGHT_PASGR_NUM><WORK_DT>20220304</WORK_DT></row><row><USE_DT>20220301</USE_DT><LINE_NUM>1</LINE_NUM><SUB_STA_NM>1</SUB_STA_NM><RIDE_PASGR_NUM>16349</RIDE_PASGR_NUM><ALIGHT_PASGR_NUM>14533</ALIGHT_PASGR_NUM><WORK_DT>20220304</WORK_DT></row><row><USE_DT>20220301</USE_DT><LINE_NUM>1</LINE_NUM><SUB_STA_NM>1</SUB_STA_NM><RIDE_PASGR_NUM>1098</RIDE_PASGR_NUM><ALIGHT_PASGR_NUM>13472</ALIGHT_PASGR_NUM><WORK_DT>20220304</WORK_DT></row></CardSubwayStatsNew>
```


6장. Node.js 12 샘플 코드

1. Node.js 다운로드 및 설치

Node.js 홈페이지에서 Node.js를 다운로드하고 설치합니다.

[Node.js 홈페이지 바로가기 >](#)

2. Node.js에 request 기능을 추가

Node.js를 실행하고 아래의 명령어를 입력합니다.

npm install --save



```
Node.js command prompt
C:\temp>npm install request --save
npm WARN deprecated har-validator@5.1.5: this library is no longer supported
npm WARN deprecated uuid@3.4.0: Please upgrade to version 7 or higher. Older versions may use Math.random() in certain
circumstances, which is known to be problematic. See https://v8.dev/blog/math-random for details.
npm WARN deprecated request@2.88.2: request has been deprecated, see https://github.com/request/request/issues/3142
added 47 packages, and audited 48 packages in 4s
2 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
```

3. 샘플 코드 및 test.js 파일 생성

메모장이나 워드패드 또는 기타 편집기를 이용하여 아래의 샘플 코드를 입력한 후, 파일명을 test.js로 저장합니다.

```
/* NodeJs 12 샘플 코드 */

var request = require('request');

var url = 'http://openapi.seoul.go.kr:8088/sample/xml/CardSubwayStatsNew/1/5/20220301';

request({
  url: url,
  method: 'GET'
}, function (error, response, body) {
  //console.log('Status', response.statusCode);
  //console.log('Headers', JSON.stringify(response.headers));
  //console.log('Reponse received', body);
});
```

4. test.js 실행 및 확인

Node.js 실행 후 아래의 명령어를 입력 및 실행하여 정상적으로 호출되는지 확인합니다.

Node test.js

```
C:\temp>node test.js
Reponse received <?xml version="1.0" encoding="UTF-8"?>
<CardSubwayStatsNew>
<list_total_count>593</list_total_count>
<RESULT>
<CODE>INFO-000</CODE>
<MESSAGE>정상 처리되었습니다</MESSAGE>
</RESULT>
<row>
<USE_DT>20220801</USE_DT>
<LINE_NUM>1호선</LINE_NUM>
```

End of Document