

2022년 IoT기반 스마트 솔루션 개발자 양성과정



Programming : Python

16-Weather Infomation

담당 교수 : 윤 종 이

010-9577-1696

oho1696@naver.com

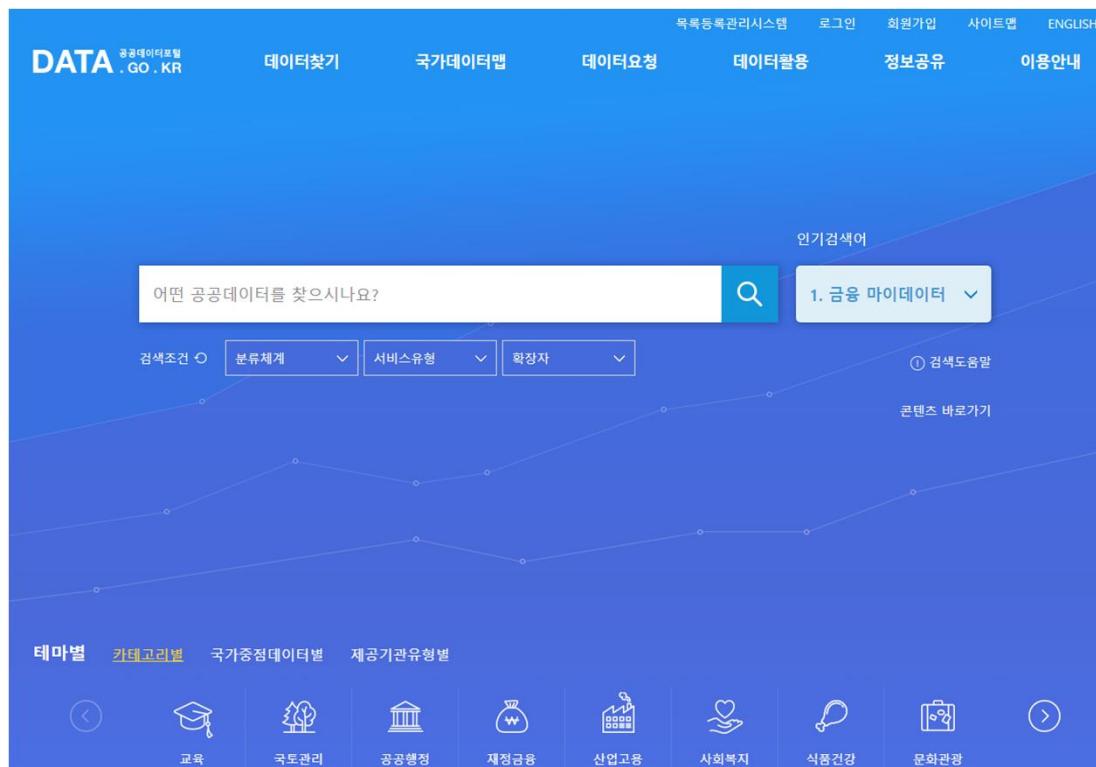
<https://cafe.naver.com/yoons2022>



충북대학교 공동훈련센터

공공데이터포털

- <https://www.data.go.kr/>



충북대학교 공동훈련센터

회원가입

회원가입



STEP1
가입확인

STEP2
개인정보수집 동의

STEP3
정보입력

STEP4
가입완료

▶ 회원 가입여부 확인을 위해 이름과 가입 시 등록하신 이메일 주소를 입력해 주세요.

* 이름

* 이메일 @
※ 이메일 주소 및 도메인은 영문, 숫자 및 특수기호(-, _, .)만 입력 가능합니다.



충북대학교 공동훈련센터

데이터찾기-동네예보, 오픈 API

오픈 API (5건)

과학기술 | 국가행정기관 | 국가증정 | 미리보기

XML | JSON 기상청_동네예보 통보문 조회서비스

발표관서나 예보구역 정보를 조건으로 기상개황, 육상예보, 해상예보를 조회하는 서비스

제공기관 기상청 수정일 2022-01-19 조회수 31373 활용신청 1875 키워드 기상개황,육상예보,해상예보 | 활용신청

과학기술 | 국가행정기관 | 국가증정 | 미리보기

XML | JSON 기상청_단기예보 ((구)_동네예보) 조회서비스

조단기실황, 조단기예보, 단기((구)동네)예보, 예보버전 정보를 조회하는 서비스입니다. 조단기실황정보는 예보 구역에 대한 대표 AWS 관측값을, 조단기예보는 예보시점부터 6시간까지...

제공기관 기상청 수정일 2022-03-31 조회수 45322 활용신청 7415 키워드 단기예보,조단기실황,조단기예보 | 활용신청

과학기술 | 국가행정기관 | 국가증정 | 미리보기

XML | JSON 기상청_서리발생 가능성 예측정보 조회서비스

동네예보자료를 활용해 서리 발생 가능성 예측 정보를 조회하는 서비스

제공기관 기상청 수정일 2020-11-23 조회수 2250 활용신청 111 키워드 서리,농작물,동네예보 | 활용신청

과학기술 | 국가행정기관 | 국가증정 | 미리보기

XML | JSON 기상청_작물별 농업주산지 상세날씨 조회서비스

농작물(36종) 주산지의 현재 날씨 및 동네예보 정보와 과거날씨 통계(일별, 순별, 월별) 정보 제공

제공기관 기상청 수정일 2021-08-04 조회수 4878 활용신청 321 키워드 기상청,기상기후,융합,주산지,주산지 기상정보 | 활용신청

과학기술 | 국가행정기관 | 국가증정 | 미리보기

XML | JSON 기상청_관광코스별 관광지 상세 날씨 조회서비스

관광코스별 관광지의 동네예보, 기상지수예보, 시군구별 관광기후지수 정보를 조회하는 서비스

제공기관 기상청 수정일 2021-08-04 조회수 11494 활용신청 1427 키워드 관광코스별 동네예보,기상지수예보,시군구별관광기후지수 | 활용신청

의견수렴
개시판



충북대학교 공동훈련센터

기상청_단기예보((구)동네예보) 조회서비스

오픈API 상세

XML JSON 기상청_단기예보 ((구)동네예보) 조회서비스

조단기실황, 조단기예보, 단기((구)동네)예보, 예보버전 정보를 조회하는 서비스입니다. 조단기실황정보는 예보 구역에 대한 대표 AWS 관측값을, 조단기예보는 예보시점부터 6시간까지의 예보를, 단기예보는 예보기간을 글피까지 확장 및 예보단위를 상세화(3시간→1시간)하여 시공간적으로 세분화한 예보를 제공합니다.

활용신청 (Red Box)

오류신고 및 담당자 문의

44 4 관심

OpenAPI 정보 메타데이터 다운로드

분류체계	과학기술 - 과학기술연구	제공기관	기상청
관리부서명	기상융합서비스과	관리부서 전화번호	042-481-7502
API 유형	REST	데이터포맷	JSON+XML
활용신청	7415	키워드	단기예보, 조단기실황, 조단기예보
등록	2021-06-28	수정	2022-03-31
심의유형	개발단계 : 허용 / 운영단계 : 허용		
비용부과유무	무료		
이용허락범위	공공저작물 출처표시		
참고문서	기상청41 단기예보 조회서비스 오픈API 활용가이드 최종.zip		



충북대학교 공동훈련센터

샘플코드

샘플코드

[Java](#)[Javascript](#)[C#](#)[PHP](#)[Curl](#)[Objective-C](#)[Python](#)[Nodejs](#)[R](#)

```
# Python3 샘플 코드 #\n\nimport requests\n\nurl = 'http://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getUltraSrtNcst'\nparams ={'serviceKey' : '서비스키', 'pageNo' : '1', 'numOfRows' : '1000', 'dataType' : 'XML', 'base_date' : '20210628', 'base_time' : '0600', 'nx' : '55', 'ny' : '127' }\n\nresponse = requests.get(url, params=params)\nprint(response.content)
```



충북대학교 공동훈련센터

마이페이지

개발계정

신청 0건 >

신청중인 단계

- 보류
- 반려

0건
0건

활용 1건 >

승인되어 활용중인 단계

- 변경신청

0건

중지0건 >

중지신청하여 운영이 중지된 단계

총1건

과학기술 기상청

활용신청 [승인] 기상청_단기예보 ((구)_동네예보) 조회서비스

신청일 2022-04-12 만료예정일 2024-04-12



충북대학교 공동훈련센터

개발계정 상세보기

기본정보

데이터명	기상청_단기예보 ((구)_동네예보) 조회서비스	상세설명	
서비스유형	REST	심의여부	자동승인
신청유형	개발계정 활용신청	처리상태	승인
활용기간	2022-04-12 ~ 2024-04-12		

서비스정보

참고문서	기상청41 단기예보 조회서비스 오픈API활용가이드 최종.zip
------	--

데이터포맷	JSON+XML
-------	----------

End Point	http://apis.data.go.kr/1360000/VilageFcstInfoService_2.0
-----------	---

API 환경 또는 API 호출 조건에 따라 인증키가 적용되는 방식이 다를 수 있습니다.

포털에서 제공되는 Encoding/Decoding 된 인증키를 적용하면서 구동되는 키를 사용하시기 바랍니다.

* 향후 포털에서 더 명확한 정보를 제공하기 위해 노력하겠습니다.

일반 인증키 (Encoding)	
일반 인증키 (Decoding)	



충북대학교 공동훈련센터

활용신청 상세기능정보

NO	상세기능	설명	일일 트래픽	미리보기
1	초단기실황조회	실황정보를 조회하기 위해 발표일자, 발표시각, 예보지점 X 좌표, 예보지점 Y 좌표의 조희 조건으로 자료구분코드, 실황값, 발표일자, 발표시각, 예보지점 X 좌표, 예보지점 Y 좌표의 정보를 조회하는 기능	10000	<button>확인</button>
2	초단기예보조회	초단기예보정보를 조회하기 위해 발표일자, 발표시각, 예보지점 X 좌표, 예보지점 Y 좌표의 조희 조건으로 자료구분코드, 예보값, 발표일자, 발표시각, 예보지점 X 좌표, 예보지점 Y 좌표의 정보를 조회하는 기능	10000	<button>확인</button>
3	단기예보조회	단기예보 정보를 조회하기 위해 발표일자, 발표시각, 예보지점 X좌표, 예보지점 Y 좌표의 조희 조건으로 발표일자, 발표시각, 자료구분문자, 예보 값, 예보일자, 예보시각, 예보지점 X 좌표, 예보지점 Y 좌표의 정보를 조회하는 기능	10000	<button>확인</button>
4	예보버전조회	단기예보정보조회서비스 각각의 오퍼레이션(초단기실황, 초단기예보, 단기예보)들의 수정된 예보버전을 파악하기 위해 예보버전을 조회하는 기능	10000	<button>확인</button>



충북대학교 공동훈련센터

Open API 활용 가이드



기상청 동네예보 조회서비스

Open API 활용 가이드

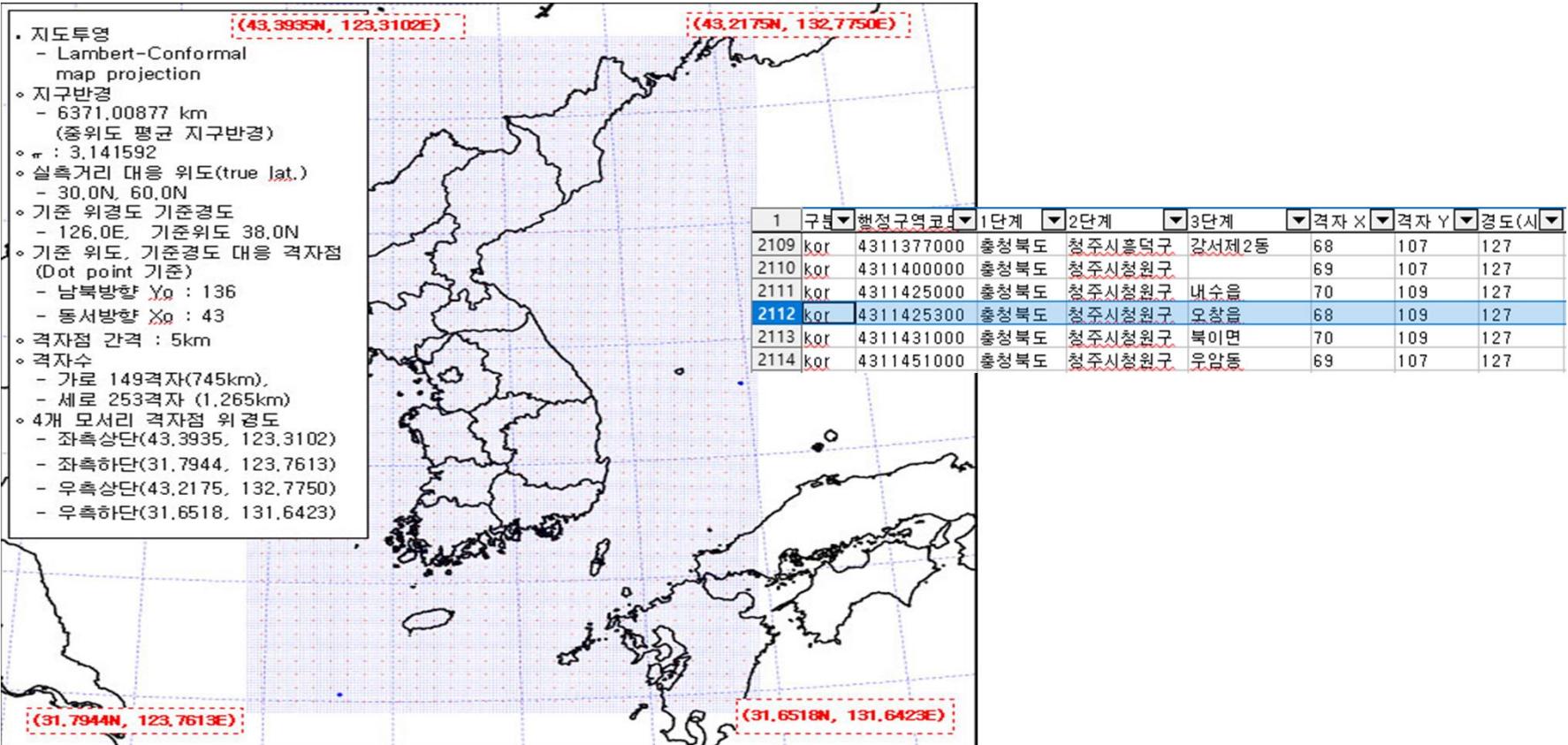
1. 서비스 명세 3

1.1 동네예보 조회서비스.....	3
가. API 서비스 개요.....	3
나. 상세기능 목록.....	4
다. 상세기능내역.....	4
1) [조단기실황조회] 상세기능명세	4
2) [조단기예보조회] 상세기능명세	7
3) [동네예보조회] 상세기능명세	10
4) [예보버전조회] 상세기능명세	13



충북대학교 공동훈련센터

격자 정보



Open API Error Code

에러코드	에러메세지	설명
00	NORMAL_SERVICE	정상
01	APPLICATION_ERROR	어플리케이션 에러
02	DB_ERROR	데이터베이스 에러
03	NODATA_ERROR	데이터없음 에러
04	HTTP_ERROR	HTTP 에러
05	SERVICETIME_OUT	서비스 연결실패 에러
10	INVALID_REQUEST_PARAMETER_ERROR	잘못된 요청 파라미터 에러
11	NO_MANDATORY_REQUEST_PARAMETERS_ERROR	필수요청 파라미터가 없음
12	NO_OPENAPI_SERVICE_ERROR	해당 오픈API서비스가 없거나 폐기됨
20	SERVICE_ACCESS_DENIED_ERROR	서비스 접근거부
21	TEMPORARILY_DISABLE_THE_SERVICEKEY_ERROR	일시적으로 사용할 수 없는 서비스 키
22	LIMITED_NUMBER_OF_SERVICE_REQUESTS_EXCEEDS_ERROR	서비스 요청제한횟수 초과에러
30	SERVICE_KEY_IS_NOT_REGISTERED_ERROR	등록되지 않은 서비스키
31	DEADLINE_HAS_EXPIRED_ERROR	기한만료된 서비스키
32	UNREGISTERED_IP_ERROR	등록되지 않은 IP
33	UNSIGNED_CALL_ERROR	서명되지 않은 호출
99	UNKNOWN_ERROR	기타에러



초단기실황조회 미리보기

NO	상세기능	설명	일일 트래픽	미리보기
1	초단기실황조회	실황정보를 조회하기 위해 발표일자, 발표시각, 예보지점 X 좌표, 예보지점 Y 좌표의 조합 조건으로 자료구분코드, 실황값, 발표일자, 발표시각, 예보지점 X 좌표, 예보지점 Y 좌표의 정보를 조회하는 기능	10000	<button>확인</button>

요청변수(Request Parameter) [닫기](#)

항목명	샘플데이터	설명
ServiceKey	NE%2B8aYywbJ%2FxVqW	공공데이터포털에서 받은 인증키
pageNo	1	페이지번호
numOfRows	1000	한 페이지 결과 수
dataType	XML	요청자료형식(XML/JSON) Default: XML
base_date	20220412	'21년 6월 28일 발표
base_time	1400	06시 발표(정시단위)
nx	68	예보지점의 X 좌표값
ny	109	예보지점의 Y 좌표값

미리보기



XML / JSON File 미리보기

This XML file does not appear to have any style information associated with it.



충북대학교 공동훈련센터

XML

XML

위키백과, 우리 모두의 백과사전.

XML(Extensible Markup Language)은 W3C에서 개발된, 다른 특수한 목적을 갖는 마크업 언어를 만드는데 사용하도록 권장하는 다목적 마크업 언어이다. XML은 SGML의 단순화된 부분집합으로, 다른 많은 종류의 데이터를 기술하는 데 사용할 수 있다. XML은 주로 다른 종류의 시스템, 특히 인터넷에 연결된 시스템끼리 데이터를 쉽게 주고 받을 수 있게 하여 HTML의 한계를 극복할 목적으로 만들어졌다.

XML은 문서를 사람과 기계 모두가 읽을 수 있는 형식을 갖도록 규정하고 있다. W3C가 만든 XML 1.0 Specification^[1]과 몇몇 다른 관련 명세들^[2]과 모든 자유 개방형 표준^[3]에서 정의되었다.

W3C는 XML 설계 목표에서 단순성과 일반성을, 그리고 인터넷을 통한 사용 가능성을 강조했다.^[4] XML은 텍스트 데이터 형식으로 유니코드를 사용해 전 세계 언어를 지원한다. XML을 설계할 때는 주로 문서를 표현하는데 집중했지만, 지금은 임의의 자료구조를 나타내는데 널리 쓰인다. 대표적인 예가 웹 서비스이다.

많은 API가 개발되어 XML 데이터를 처리하고자 하는 소프트웨어 개발자들이 활용하고 있다. 또한, 여러 가지 스키마 시스템이 있어서 XML 기반 언어의 정의를 보다 쉽게 할 수 있도록 도와준다.

XML

```
<?xml version="1.0"?>
<quiz>
  <qanda seq="1">
    <question>
      Who was the forty-second
      president of the U.S.A.?
    </question>
    <answer>
      William Jefferson Clinton
    </answer>
  </qanda>
  <!-- Note: We need to add
       more questions later.-->
</quiz>
```

XML

확장자

.xml

MIME 종류

application/xml

개발

W3C

파일 포맷 종류

마크업 언어

웹사이트

<http://www.w3c.org>



충북대학교 공동훈련센터

XML Structure

```
1 <?xml version="1.0" encoding="1.0" standalone="yes" ?>
2
3 <root>
4   <child>
5     <subchild>내</subchild>
6   </child>
7 </root>
8
```

```
<body>
  - <items>
    - <item>
      <baseDate>20181013</baseDate>
      <baseTime>1800</baseTime>
      <category>PTY</category>
      <nx>68</nx>
      <ny>109</ny>
      <obsrValue>0</obsrValue>
    </item>
```



Response XML

```
<?xml version="1.0" encoding="UTF-8" standalone="true"?>
- <response>
  - <header>
    <resultCode>0000</resultCode>
    <resultMsg>OK</resultMsg>
  </header>
  - <body>
    - <items>
      - <item>
        <baseDate>20181012</baseDate>
        <baseTime>1700</baseTime>
        <category>PTY</category>
        <nx>68</nx>
        <ny>109</ny>
        <obsrValue>0</obsrValue>
      </item>
      - <item>
        <baseDate>20181012</baseDate>
        <baseTime>1700</baseTime>
        <category>REH</category>
        <nx>68</nx>
        <ny>109</ny>
        <obsrValue>71</obsrValue>
      </item>
      - <item>
        <baseDate>20181012</baseDate>
        <baseTime>1700</baseTime>
        <category>RN1</category>
        <nx>68</nx>
        <ny>109</ny>
        <obsrValue>0</obsrValue>
      </item>
      - <item>
        <baseDate>20181012</baseDate>
        <baseTime>1700</baseTime>
        <category>T1H</category>
        <nx>68</nx>
        <ny>109</ny>
        <obsrValue>11</obsrValue>
      </item>
    </items>
  </body>
</response>
```

```
- <item>
  <baseDate>20181012</baseDate>
  <baseTime>1700</baseTime>
  <category>UUU</category>
  <nx>68</nx>
  <ny>109</ny>
  <obsrValue>0</obsrValue>
</item>
- <item>
  <baseDate>20181012</baseDate>
  <baseTime>1700</baseTime>
  <category>VEC</category>
  <nx>68</nx>
  <ny>109</ny>
  <obsrValue>0</obsrValue>
</item>
- <item>
  <baseDate>20181012</baseDate>
  <baseTime>1700</baseTime>
  <category>VVV</category>
  <nx>68</nx>
  <ny>109</ny>
  <obsrValue>0</obsrValue>
</item>
- <item>
  <baseDate>20181012</baseDate>
  <baseTime>1700</baseTime>
  <category>WSD</category>
  <nx>68</nx>
  <ny>109</ny>
  <obsrValue>0</obsrValue>
</item>
</items>
<numOfRows>10</numOfRows>
<pageNo>1</pageNo>
<totalCount>8</totalCount>
</body>
</response>
```



json.org

- <https://www.json.org>

The screenshot shows the json.org homepage. At the top left is a large black 'O' logo. To its right is a white rectangular box containing the Korean text "JSON 개요". Below the logo and box is a horizontal line of language links in various languages. Underneath is the text "ECMA-404 The JSON Data Interchange Standard.". A large paragraph defines JSON as a data interchange notation used in programming languages like JavaScript. Below this, a bulleted list details the two main structures: objects and arrays. To the right, a diagram illustrates the JSON grammar with terms like "json element", "value", "object", "array", "string", "number", "true", "false", and "null". It also shows the recursive definition of an object as a pair of curly braces containing whitespace and members.

json.org

JSON 개요

Български 中文 Český Dansk Nederlands English Esperanto Français Deutsch Ελληνικά עברית Magyar Indonesia
Italiano 日本 한국어 فارسی Polski Português Română Русский Српско-хрватски Slovenčina Español Svenska Türkçe Tiếng Việt

ECMA-404 The JSON Data Interchange Standard.

JSON (JavaScript Object Notation)은 경량의 DATA-교환 형식이다. 이 형식은 사람이 읽고 쓰기에 용이하며, 기계가 분석하고 생성함에도 용이하다. JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999의 일부에 토대를 두고 있다. JSON은 완벽하게 언어로 부터 독립적이지만 C-family 언어 - C, C++, C#, Java, JavaScript, Perl, Python 그외 다수 - 의 프로그래머들에게 친숙한 관습을 사용하는 텍스트 형식이다. 이러한 속성들이 JSON을 이상적인 DATA-교환 언어로 만들고 있다.

JSON은 두개의 구조를 기본으로 두고 있다:

- name/value 형태의 쌍으로 collection 타입. 다양한 언어들에서, 이는 object, record, struct(구조체), dictionary, hash table, 키가 있는 list, 또는 연상 배열로서 실현 되었다.
- 값들의 순서화된 리스트. 대부분의 언어들에서, 이는 array, vector, list, 또는 sequence로서 실현 되었다.

json element

value

object

array

string

number

"true"

"false"

"null"

object

{ ws }

{ members }

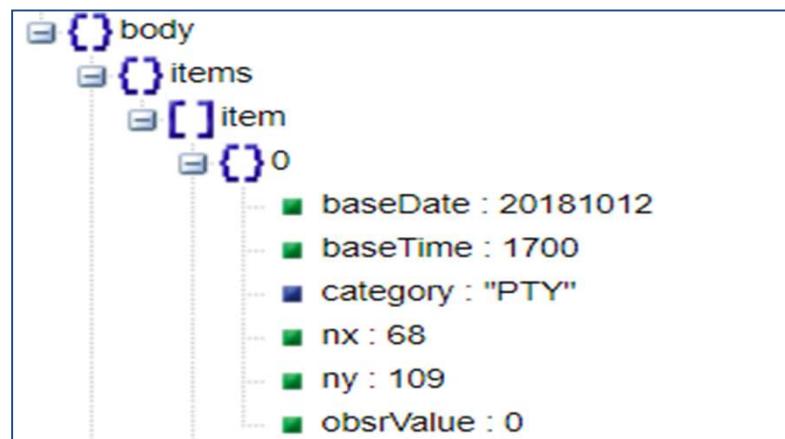
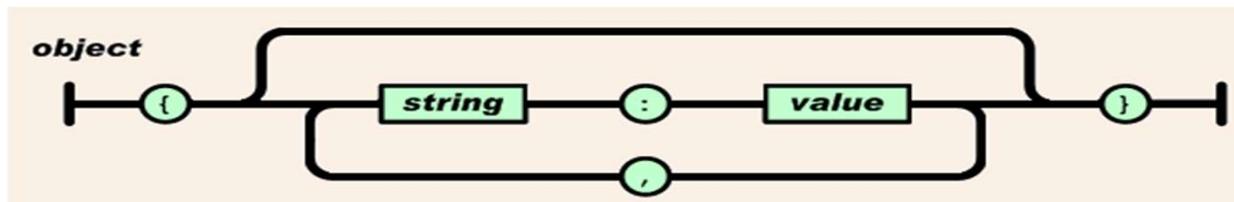
members

member

member , members



Json Structure



1.초단기실황조회

요청변수(Request Parameter)		
항목명	샘플데이터	설명
ServiceKey	NE%2B8aYywbiJ%2FxVqW	공공데이터포털에서 받은 인증키
pageNo	1	페이지번호
numOfRows	1000	한 페이지 결과 수
dataType	XML	요청자료형식(XML/JSON) Default: XML
base_date	20220412	'21년 6월 28일 발표
base_time	1400	06시 발표(정시단위)
nx	68	예보지점의 X 좌표값
ny	109	예보지점의 Y 좌표값

[미리보기](#)



충북대학교 공동훈련센터

요청/응답 메시지

- [https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getUltraSrtNcst?
serviceKey=인증키&pageNo=1&numOfRows=1000&dataType=XML&base_date=20220412
&base_time=1400&nx=68&ny=109](https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getUltraSrtNcst?serviceKey=인증키&pageNo=1&numOfRows=1000&dataType=XML&base_date=20220412&base_time=1400&nx=68&ny=109)

T1H	기온	°C	27.5
RN1	1시간 강수량	mm	0
UUU	동서바람성분	m/s	1.8
VVV	남북바람성분	m/s	3.3
REH	습도	%	45
PTY	강수형태	코드값	0
VEC	풍향	deg	209
WSD	풍속	m/s	3.8

```
{"response": {"header": {"resultCode": "00", "resultMsg": "NORMAL_SERVICE"}, "body": {"dataType": "JSON", "items": {"item": [{"baseDate": "20220412", "baseTime": "1400", "category": "PTY", "nx": 68, "ny": 109, "obsrValue": "0"}, {"baseDate": "20220412", "baseTime": "1400", "category": "REH", "nx": 68, "ny": 109, "obsrValue": "43"}, {"baseDate": "20220412", "baseTime": "1400", "category": "RNI", "nx": 68, "ny": 109, "obsrValue": "0"}, {"baseDate": "20220412", "baseTime": "1400", "category": "T1H", "nx": 68, "ny": 109, "obsrValue": "28.1"}, {"baseDate": "20220412", "baseTime": "1400", "category": "UUU", "nx": 68, "ny": 109, "obsrValue": "1.9"}, {"baseDate": "20220412", "baseTime": "1400", "category": "VEC", "nx": 68, "ny": 109, "obsrValue": "203"}, {"baseDate": "20220412", "baseTime": "1400", "category": "VVW", "nx": 68, "ny": 109, "obsrValue": "4.4"}, {"baseDate": "20220412", "baseTime": "1400", "category": "WSD", "nx": 68, "ny": 109, "obsrValue": "4.8"}]}, {"pageNo": 1, "numOfRows": 1000, "totalCount": 8}]}}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <header>
    <resultCode>00</resultCode>
    <resultMsg>NORMAL_SERVICE</resultMsg>
  </header>
  <body>
    <dataType>XML</dataType>
    <items>
      <item>
        <baseDate>20220412</baseDate>
        <baseTime>1400</baseTime>
        <category>PTV</category>
        <nx>68</nx>
        <ny>109</ny>
        <obsrValue>0</obsrValue>
      </item>
      <item>
        <baseDate>20220412</baseDate>
        <baseTime>1400</baseTime>
        <category>REH</category>
        <nx>68</nx>
        <ny>109</ny>
        <obsrValue>45</obsrValue>
      </item>
      <item>
        <baseDate>20220412</baseDate>
        <baseTime>1400</baseTime>
        <category>RNI</category>
        <nx>68</nx>
        <ny>109</ny>
        <obsrValue>0</obsrValue>
      </item>
      <item>
        <baseDate>20220412</baseDate>
        <baseTime>1400</baseTime>
        <category>THC</category>
        <nx>68</nx>
        <ny>109</ny>
        <obsrValue>27.5</obsrValue>
      </item>
      <item>
        <baseDate>20220412</baseDate>
        <baseTime>1400</baseTime>
        <category>UUU</category>
        <nx>68</nx>
        <ny>109</ny>
        <obsrValue>1.8</obsrValue>
      </item>
      <item>
        <baseDate>20220412</baseDate>
        <baseTime>1400</baseTime>
        <category>VEC</category>
        <nx>68</nx>
        <ny>109</ny>
        <obsrValue>209</obsrValue>
      </item>
    </items>
  </body>
</response>
```



2.초단기예보조회

요청변수(Request Parameter)

[닫기](#)

항목명	샘플데이터	설명
ServiceKey	NE%2B8aYywbiJ%2FxVqW	공공데이터포털에서 받은 인증키
pageNo	1	페이지번호
numOfRows	1000	한 페이지 결과 수
dataType	XML	요청자료형식(XML/JSON) Default: XML
base_date	20220412	'21년 6월 28일 발표
base_time	1400	06시30분 발표(30분 단위)
nx	68	예보지점 X 좌표값
ny	109	예보지점 Y 좌표값



요청/응답 메시지

- [https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getUltraSrtFcst?
serviceKey=인증키 &pageNo=1&numOfRows=1000&dataType=XML&base_date=20220412
&base_time=1400&nx=68&ny=109](https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getUltraSrtFcst?serviceKey=인증키&pageNo=1&numOfRows=1000&dataType=XML&base_date=20220412&base_time=1400&nx=68&ny=109)

T1H	기온	°C	10
RN1	1시간 강수량	범주 (1 mm)	8
SKY	하늘상태	코드값	4
UUU	동서바람성분	m/s	12
VVV	남북바람성분	m/s	12
REH	습도	%	8
PTY	강수형태	코드값	4
LGT	낙뢰	코드값	4
VEC	풍향	deg	10
WSD	풍속	m/s	10

```
<response>
  <header>
    <resultCode>00</resultCode>
    <resultMsg>NORMAL_SERVICE</resultMsg>
  </header>
  <body>
    <data type="XML"></data>
    <items>
      <item>
        <baseDate>20220412</baseDate>
        <baseTime>1430</baseTime>
        <category>LGT</category>
        <fcstDate>20220412</fcstDate>
        <fcstTime>1500</fcstTime>
        <fcstValue>0</fcstValue>
        <npx>68</npx>
        <ny>109</ny>
      </item>
      <item>
        <baseDate>20220412</baseDate>
        <baseTime>1430</baseTime>
        <category>LGT</category>
        <fcstDate>20220412</fcstDate>
        <fcstTime>1600</fcstTime>
        <fcstValue>0</fcstValue>
        <npx>68</npx>
        <ny>109</ny>
      </item>
      <item>
        <baseDate>20220412</baseDate>
        <baseTime>1430</baseTime>
        <category>LGT</category>
        <fcstDate>20220412</fcstDate>
        <fcstTime>1700</fcstTime>
        <fcstValue>0</fcstValue>
        <npx>68</npx>
        <ny>109</ny>
      </item>
      <item>
        <baseDate>20220412</baseDate>
        <baseTime>1430</baseTime>
        <category>LGT</category>
        <fcstDate>20220412</fcstDate>
        <fcstTime>1800</fcstTime>
        <fcstValue>0</fcstValue>
        <npx>68</npx>
        <ny>109</ny>
      </item>
      <item>
        <baseDate>20220412</baseDate>
        <baseTime>1430</baseTime>
        <category>LGT</category>
        <fcstDate>20220412</fcstDate>
        <fcstTime>1900</fcstTime>
        <fcstValue>0</fcstValue>
        <npx>68</npx>
        <ny>109</ny>
      </item>
    </items>
  </body>
</response>
```



3. 단기예보조회

요청변수(Request Parameter)			닫기
항목명	샘플데이터	설명	
ServiceKey	NE%2B8aYywbiJ%2FxVqW	공공데이터포털에서 받은 인증키	
pageNo	1	페이지번호	
numOfRows	1000	한 페이지 결과 수	
dataType	XML	요청자료형식(XML/JSON) Default: XML	
base_date	20220412	'21년 6월 28일 발표	
base_time	1400	05시 발표	
nx	68	예보지점의 X 좌표값	
ny	109	예보지점의 Y 좌표값	

[미리보기](#)



요청/응답 메시지

- [https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getVilageFcst?
serviceKey=인증키&pageNo=1&numOfRows=1000&dataType=XML&base_date=20220412
&base_time=1400&nx=68&ny=109](https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getVilageFcst?serviceKey=인증키&pageNo=1&numOfRows=1000&dataType=XML&base_date=20220412&base_time=1400&nx=68&ny=109)

POP	강수확률	%
PTY	강수형태	코드값
PCP	1시간 강수량	범주 (1 mm)
REH	습도	%
SNO	1시간 신적설	범주(1 cm)
SKY	하늘상태	코드값
TMP	1시간 기온	°C
TMN	일 최저기온	°C
TMX	일 최고기온	°C
UUU	풍속(동서성분)	m/s
VVV	풍속(남북성분)	m/s
WAV	파고	M
VEC	풍향	deg
WSD	풍속	m/s

```
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <header>
    <resultCode>00</resultCode>
    <resultMsg>NORMAL_SERVICE</resultMsg>
  </header>
  <body>
    <dataType>XML</dataType>
    <items>
      <item>
        <baseDate>20220412</baseDate>
        <baseTime>1400</baseTime>
        <category>TMR</category>
        <fcstDate>20220412</fcstDate>
        <fcstTime>1500</fcstTime>
        <fcstValue>26</fcstValue>
        <nx>68</nx>
        <ny>109</ny>
      </item>
      <item>
        <baseDate>20220412</baseDate>
        <baseTime>1400</baseTime>
        <category>UUU</category>
        <fcstDate>20220412</fcstDate>
        <fcstTime>1500</fcstTime>
        <fcstValue>2.2</fcstValue>
        <nx>68</nx>
        <ny>109</ny>
      </item>
      <item>
        <baseDate>20220412</baseDate>
        <baseTime>1400</baseTime>
        <category>VVV</category>
        <fcstDate>20220412</fcstDate>
        <fcstTime>1500</fcstTime>
        <fcstValue>1.5</fcstValue>
        <nx>68</nx>
        <ny>109</ny>
      </item>
      <item>
        <baseDate>20220412</baseDate>
        <baseTime>1400</baseTime>
        <category>VEC</category>
        <fcstDate>20220412</fcstDate>
        <fcstTime>1500</fcstTime>
        <fcstValue>236</fcstValue>
        <nx>68</nx>
        <ny>109</ny>
      </item>
      <item>
        <baseDate>20220412</baseDate>
        <baseTime>1400</baseTime>
        <category>WSD</category>
        <fcstDate>20220412</fcstDate>
        <fcstTime>1500</fcstTime>
        <fcstValue>2.7</fcstValue>
        <nx>68</nx>
        <ny>109</ny>
      </item>
    </items>
  </body>
</response>
```



4. 예보버전조회

요청변수(Request Parameter)			닫기
항목명	샘플데이터	설명	
ServiceKey	NE%2B8aYywbiJ%2FxVqW	공공데이터포털에서 받은 인증키	
pageNo	1	페이지번호	
numOfRows	1000	한 페이지 결과 수	
dataType	XML	요청자료형식(XML/JSON) Default: XML	
ftype	ODAM	파일구분 -ODAM: 동네예보실황 -VSRT: 동네예보초단기 -SHRT: 동네예보단기	
basedatetime	202204121400	각각의 base_time 로 검색	

[미리보기](#)



요청/응답 메시지

- https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getFcst
Version?serviceKey=인증키
&pageNo=1&numOfRows=1000&dataType=XML&ftype=ODAM
&basedatetime=202204121400

```
{"response":{"header":{"resultCode":"00","resultMsg":"NORMAL_SERVICE"},"body":{"dataType":"JSON","items":[{"item":{"filetype":"ODAM","version":"20220412152332"}]},"pageNo":1,"numOfRows":1000,"totalCount":1}}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<response>
    <header>
        <resultCode>00</resultCode>
        <resultMsg>NORMAL_SERVICE</resultMsg>
    </header>
    <body>
        <dataType>XML</dataType>
        <items>
            <item>
                <filetype>ODAM</filetype>
                <version>20220412152332</version>
            </item>
        </items>
        <numOfRows>1000</numOfRows>
        <pageNo>1</pageNo>
        <totalCount>1</totalCount>
    </body>
</response>
```



Ex1 : 초단기실황 : Weather-1.py

The screenshot shows the Thonny IDE interface. The top bar displays the path: D:\AworkCom\학사업무\2022학사\충북대학교\Raspberry\program\weather-1.py. The main window contains the Python code for retrieving weather data from a specific URL. The code uses the `urllib.request` module to send a GET request to the URL, which includes parameters like serviceKey, numOfRow, pageNo, dataType, base_date, base_time, nx, and ny. The response is read and decoded as UTF-8, and then printed. Below the code editor is a shell window showing the command `%Run weather-1.py` and the resulting XML response. The XML response includes header information, a resultCode of 00, and a resultMsg of "NORMAL_SERVICE". The bottom right corner of the shell window indicates Python 3.7.9.

```
1 import urllib.request
2 import json
3
4 url='https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getUltraSrtNcst'
5 serviceKey='ServiceKey=' + 'NE%2B8aYywbij%2FxVqW7Al407gl1QkndZfcI3WDuErzmlW2oYSdZC
6 numOfRow='&numOfRows=1000'
7 pageNo='&pageNo=1'
8 dataType='&dataType=XML'
9 base_date='&base_date=' + '20220412'
10 base_time='&base_time=' + '1200'
11 nx='&nx=68'
12 ny='&ny=109'
13
14 queryParams='?' + serviceKey + numOfRow + pageNo + dataType + base_date + base_time + nx + ny
15 response=urllib.request.urlopen(url+queryParams).read().decode('utf8')
16 print (response)
17
```

```
>>> %Run weather-1.py
<?xml version="1.0" encoding="UTF-8"?>
<response><header><resultCode>00</resultCode><resultMsg>NORMAL_SERVICE ...
```

Python 3.7.9



Ex1 : Run XML

Squeezed text (1338 characters)

For performance reasons, Shell avoids showing very long lines in full (see Tools => Options => Shell). Here you can interact with the original text fragment.

Wrap text (may be slow)

```
<response><header><resultCode>00</resultCode><resultMsg>NORMAL_SERVICE</resultMs g></header><body><dataType>XML</dataType><items><item><baseDate>20210512</baseDa te><baseTime>0900</baseTime><category>PTY</category><nx>68</nx><ny>109</ny><obsr Value>0</obsrValue></item><item><baseDate>20210512</baseDate><baseTime>0900</bas eTime><category>REH</category><nx>68</nx><ny>109</ny><obsrValue>31</obsrValue></item><item><baseDate>20210512</baseDate><baseTime>0900</baseTime><category>RN1</category><nx>68</nx><ny>109</ny><obsrValue>0</obsrValue></item><item><baseDate>20210512</baseDate><baseTime>0900</baseTime><category>T1H</category><nx>68</nx><n y>109</ny><obsrValue>21.8</obsrValue></item><item><baseDate>20210512</baseDate>< baseTime>0900</baseTime><category>UUU</category><nx>68</nx><ny>109</ny><obsrValu
```

Shell

```
>>> %Run weather-1.py
```

```
<?xml version="1.0" encoding="UTF-8"?>
<response><header><resultCode>00</resultCode><resultMsg>NORMAL_SERVICE</resultM sg></header><body><dataType>XML</dataType><items><item><baseDate>20210512</ba seDate><baseTime>0900</baseTime><category>PTY</category><nx>68</nx><ny>109</ny ><obsrValue>0</obsrValue></item><item><baseDate>20210512</baseDate><baseTime>0 900</baseTime><category>REH</category><nx>68</nx><ny>109</ny><obsrValue>31</ob srValue></item><item><baseDate>20210512</baseDate><baseTime>0900</baseTime><category>RN1</category><nx>68</nx><ny>109</ny><obsrValue>0</obsrValue></item><item><baseDate>20210512</baseDate><baseTime>0900</baseTime><category>T1H</category><nx>68</nx><ny>109</ny><obsrValue>21.8</obsrValue></item><item><baseDate>20210512</baseDate><baseTime>0900</baseTime><category>UUU</category><nx>68</nx><ny>109</ny><obsrValue>-0.7</obsrValue></item><item><baseDate>20210512</baseDat e><baseTime>0900</baseTime><category>VEC</category><nx>68</nx><ny>109</ny><ob srValue>122</obsrValue></item><item><baseDate>20210512</baseDate><baseTime>0900 </baseTime><category>VVV</category><nx>68</nx><ny>109</ny><obsrValue>0.5</ob srValue></item><item><baseDate>20210512</baseDate><baseTime>0900</baseTime><category>WSD</category><nx>68</nx><ny>109</ny><obsrValue>1</obsrValue></item><item><numOfRows>10</numOfRows><pageNo>1</pageNo><totalCount>8</totalCount></body ></response>
```



Ex1 : Run JSON

Shell

```
>>> %Run weather-1.py
```

```
{"response":{"header":{"resultCode":"00","resultMsg":"NORMAL_SERVICE"},"body": {"dataType":"JSON","items":{"item":[{"baseDate":"20210512","baseTime":"0900","category":"PTY","nx":68,"ny":109,"obsrValue":"0"}, {"baseDate":"20210512","baseTime":"0900","category":"REH","nx":68,"ny":109,"obsrValue":"31"}, {"baseDate":"20210512","baseTime":"0900","category":"RN1","nx":68,"ny":109,"obsrValue":"0"}, {"baseDate":"20210512","baseTime":"0900","category":"T1H","nx":68,"ny":109,"obsrValue":"21.8"}, {"baseDate":"20210512","baseTime":"0900","category":"UUU","nx":68,"ny":109,"obsrValue": "-0.7"}, {"baseDate":"20210512","baseTime":"0900","category":"VEC","nx":68,"ny":109,"obsrValue":"122"}, {"baseDate":"20210512","baseTime":"0900","category":"VVV","nx":68,"ny":109,"obsrValue":"0.5"}, {"baseDate":"20210512","baseTime":"0900","category":"WSD","nx":68,"ny":109,"obsrValue":"1"}],"pageNo":1,"numOfRows":10,"totalCount":8}}}
```

```
>>>
```



충북대학교 공동훈련센터

Ex2 : weather-2.py

```
weather-2.py x
1 import urllib.request
2 import json
3 from time import localtime,strftime
4 import datetime
5
6 url='https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getUltraSrtNcst'
7 serviceKey='ServiceKey=' + 'NE%2B8aYywbij%2FxVqW7Al407gl1QkndZfcI3WDuErzmlW2oYSdZC'
8 numOfRow='&numOfRows=10'
9 pageNo='&pageNo=1'
10 dataType='&dataType=JSON'
11
12 current=datetime.datetime.now()
13 base_date='&base_date=' + current.strftime("%Y%m%d")
14 if localtime().tm_min>30:
15     base_time='&base_time=' + current.strftime("%H00")
16 else:
17     current=current-datetime.timedelta(hours=1)
18     base_time='&base_time=' + current.strftime("%H00")
19
20 nx='&nx=68'
21 ny='&ny=109'
22
23 queryParams='?' + serviceKey+numOfRow+pageNo+dataType+base_date+base_time+nx+ny
24 response=urllib.request.urlopen(url+queryParams).read().decode('utf8')
25
26 print (response)
```



Ex2 : Run JSON

```
Shell x
>>> %Run weather-2.py

{"response": {"header": {"resultCode": "00", "resultMsg": "NORMAL_SERVICE"}, "body": {"data": [{"category": "PTY", "date": "20220412", "time": "1600", "nx": 68, "ny": 109, "obsrValue": "0"}, {"category": "REH", "date": "20220412", "time": "1600", "nx": 68, "ny": 109, "obsrValue": "36"}, {"category": "RN1", "date": "20220412", "time": "1600", "nx": 68, "ny": 109, "obsrValue": "0"}, {"category": "T1H", "date": "20220412", "time": "1600", "nx": 68, "ny": 109, "obsrValue": "27.2"}, {"category": "UUU", "date": "20220412", "time": "1600", "nx": 68, "ny": 109, "obsrValue": "3.8"}, {"category": "VEC", "date": "20220412", "time": "1600", "nx": 68, "ny": 109, "obsrValue": "229"}, {"category": "VVV", "date": "20220412", "time": "1600", "nx": 68, "ny": 109, "obsrValue": "3.3"}, {"category": "WSD", "date": "20220412", "time": "1600", "nx": 68, "ny": 109, "obsrValue": "5"}], "pageNo": 1, "totalCount": 8}}}

>>>
```



Ex3 : weather-3.py

```
weather-3.py * 
1 import urllib.request
2 import json
3 from time import localtime,strftime
4 import datetime
5
6 url='https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getUltraSrtNcst'
7 serviceKey='ServiceKey=' + 'NE%2B8aYywbiJ%2FxVqW7Al407gl1QkndZfcI3WDuErzmW2oYSdZCt
8 numOfRow='&numOfRows=10'
9 pageNo='&pageNo=1'
10 dataType='&dataType=JSON'
11
12 current=datetime.datetime.now()
13 base_date='&base_date=' + current.strftime("%Y%m%d")
14 if localtime().tm_min>30:
15     base_time='&base_time=' + current.strftime("%H00")
16 else:
17     current=current-datetime.timedelta(hours=1)
18     base_time='&base_time=' + current.strftime("%H00")
19
20 nx='&nx=68'
21 ny='&ny=109'
22
23 queryParams='?' + serviceKey+numOfRow+pageNo+dataType+base_date+base_time+nx+ny
24 response=urllib.request.urlopen(url+queryParams).read().decode('utf8')
25
26 weather=json.loads(response)
27 totalCount=weather["response"]["body"]["totalCount"]
28 for k in range(totalCount):
29     category=weather["response"]["body"]["items"]["item"][k]["category"]
30     obsrValue=weather["response"]["body"]["items"]["item"][k]["obsrValue"]
31     print (category," : ",obsrValue)
```



Ex3 : Run json parsing

```
Shell >>> %Run weather-3.py
&base_time=1600
PTY   : 0
REH   : 37
RN1   : 0
T1H   : 27.6
UUU   : 3.7
VEC   : 224
VVV   : 3.8
WSD   : 5.3
```



Ex4 : weather-4.py

```
weather-4.py x
1 import urllib.request
2 import json
3 from time import localtime,strftime
4 import datetime
5
6 url='https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getUltraSrtNcst'
7 serviceKey='ServiceKey=' + 'NE%2B8aYywbiJ%2FxVqW7Al407gl1QkndZfcI3WDuErzmW2oYSdZCt
8 numOfRow='&numOfRows=10'
9 pageNo='&pageNo=1'
10 dataType='&dataType=JSON'
11
12 current=datetime.datetime.now()
13 base_date='&base_date=' + current.strftime("%Y%m%d")
14 if localtime().tm_min>30:
15     base_time='&base_time=' + current.strftime("%H00")
16 else:
17     current=current-datetime.timedelta(hours=1)
18     base_time='&base_time=' + current.strftime("%H00")
19
20 nx='&nx=68'
21 ny='&ny=109'
22
23 queryParams='?' + serviceKey+numOfRow+pageNo+dataType+base_date+base_time+nx+ny
24 #print(queryParams)
25 response=urllib.request.urlopen(url+queryParams).read().decode('utf8')
26
27 weather=json.loads(response)
28
29 for k in weather["response"]["body"]["items"]["item"]:
30     print (k["category"], " : ", k["obsrValue"])
```



Ex4 : Run json parsing

```
Shell ×
>>> %Run weather-4.py
PTY    : 0
REH    : 37
RN1    : 0
T1H    : 27
UUU    : 3.7
VEC    : 227
VVV    : 3.5
WSD    : 5.1
>>>
```



Code Index

예보구분	항목값	항목명	단위	Bit수
초단기실황	T1H	기온	°C	10
	RN1	1시간 강수량	mm	8
	UUU	동서바람성분	m/s	12
	VVV	남북바람성분	m/s	12
	REH	습도	%	8
	PTY	강수형태	코드값	4
	VEC	풍향	0	10
	WSD	풍속	1	10



Ex5 : weather-5.py

```
weather-5.py x
1 import urllib.request
2 import json
3 from time import localtime,strftime
4 import datetime
5
6 dic={"T1H":"Temperature","RN1":"Rainfall","UUU":"Wind E-W",
7      "VVV":"Wind S-N","REH":"Humidity","PTY":"Precipitation type",
8      "VEC":"Wind direction","WSD":"Wind speed"}
9
10 url='https://apis.data.go.kr/1360000/VillageFcstInfoService_2.0/getUltraSrtNcst'
11 serviceKey='ServiceKey=' + 'NE%2B8aYywbij%2FxVqW7Al407gl1QkndZfcI3WDuErzmW2oYSdZCt
12 numOfRow='&numOfRows=10'
13 pageNo='&pageNo=1'
14 dataType='&dataType=JSON'
15
16 current=datetime.datetime.now()
17 base_date='&base_date=' + current.strftime("%Y%m%d")
18 if localtime().tm_min>30:
19     base_time='&base_time=' + current.strftime("%H00")
20 else:
21     current=current-datetime.timedelta(hours=1)
22     base_time='&base_time=' + current.strftime("%H00")
23
24 nx='&nx=68'
25 ny='&ny=109'
26
27 queryParams='?' + serviceKey+numOfRow+pageNo+dataType+base_date+base_time+nx+ny
28 response=urllib.request.urlopen(url+queryParams).read().decode('utf8')
29
30 weather=json.loads(response)
31 for k in weather["response"]["body"]["items"]["item"]:
32     print (dic[k["category"]]," : ", k["obsrValue"])
```



Ex5 : Run Dictionary

```
Shell x
>>> %Run weather-5.py

Precipitation type  :  0
Humidity    :  36
Rainfall     :  0
Temperature   :  27.2
Wind E-W      :  3.8
Wind direction :  229
Wind S-N      :  3.3
Wind speed    :  5

>>>
```

