

API 데이터 자동화 연동 및 처리

강사: 박은지

강사 소개



박은지

- AI/Data 분야 강사
- (전) AI/Data 분야 콘텐츠 기획 및 제작 PD

커리큘럼 소개

[8일차]

- API와 RESTful API
- 공공 데이터와 Open API
- JSON의 구조
- JSON 파싱 및 데이터 추출
- 기상 API 데이터 전처리 및 저장
- 실시간 기상 API 데이터 전처리 및 저장 자동화

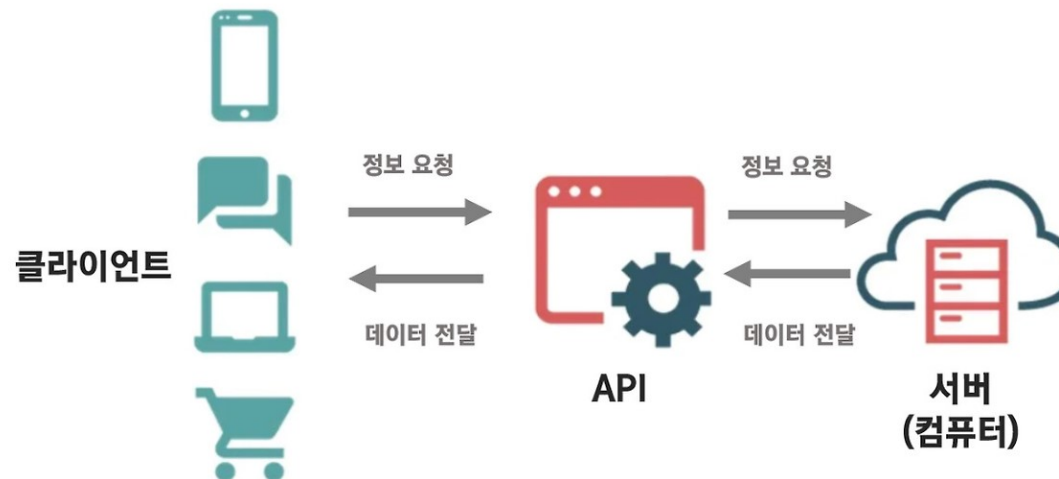
API와 RESTful API

- API
- REST API
- RESTful API
- HTTP 메소드(실습)

API란?

API(Application Programing Interface)

- 애플리케이션 간 또는 외부 시스템과 통신하기 위한 규칙 또는 인터페이스



API란?

장점

- 빠른 개발 속도
 - 기존 기능 재사용
- 유지보수 용이
 - 하나의 API 서버에서 다양한 기능이 제공되므로 데이터가 일관되고 기능 업데이트와 유지보수가 용이
- 호환성
 - 다양한 시스템 간의 호환성 보장

API란?

장점

- 확장성
 - 버전 분리 등을 통해 안정적인 확장 가능
- 보안성과 접근 제어
 - API 키, 토큰 등을 이용해 권한 관리 및 로그 추적 가능
- 자동화
 - 코드로 API를 호출해 자동화 가능

API란?

종류

■ 웹 API

- 웹 서비스의 상호작용을 위한 API, 주로 HTTP/HTTPS 프로토콜 사용

- 주로 JSON 형식으로 요청/응답 처리
- 웹 서비스 간 데이터 교환에 사용

■ 운영 체제 API

- 운영 체제의 내부 기능에 접근하기 위한 API
- 예: 파일 시스템 접근, 프로세스 관리

API란?

웹 API의 대표적 유형

- RESTful API
 - REST 아키텍처 스타일을 따르는 가장 널리 사용되는 웹 API
 - 직관적이고 가볍고 빠름
 - 데이터 형식은 JSON, XML, 텍스트 등
- SOAP(Simple Object Access Protocol) API
 - XML 기반 메시지 사용, 복잡한 구조
 - 무겁지만 표준화된 API
 - 보안, 인증, 트랜잭션 처리에 유리
 - 금융, 공공기관 등에서 사용

API란?

SOAP XML 메시지 예시

```
<soap:Envelope  
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">  
  <soap:Body>  
    <GetWeather>  
      <city>Seoul</city>  
    </GetWeather>  
  </soap:Body>  
</soap:Envelope>
```

HTTP 요청과 응답

HTTP(HyperText Transfer Protocol)

- 웹에서 클라이언트와 서버가 데이터를 주고 받는 통신 규약
- API와의 차이점
 - HTTP: 통신 규약, 데이터 전송 방식(형식과 방법) 정의
 - API: 기능 인터페이스, 어떤 기능을 제공하고 어떻게 호출하는지 설명(약속)

HTTP 요청과 응답

HTTP 요청(Request)

- 클라이언트가 서버에 데이터를 요청할 때 보내는 메시지
- 요청 구조
 - 요청 라인: 작업 종류
 - 요청 헤더: 요청 정보와 조건
 - 요청 바디: 서버에 보낼 데이터(POST, PUT일 때 사용)

HTTP 요청과 응답

HTTP 요청(Request)

- 클라이언트가 서버에 데이터를 요청할 때 보내는 메시지

```
POST /login HTTP/1.1
```

← 요청 라인(동작, 자원, 버전)

```
Host: example.com
```

```
Content-Type: application/json
```

← 요청 헤더(데이터 형식, 인증 정보)

```
Authorization: Bearer abcdef123456
```

```
{
```

← 요청 바디(실제 데이터)

```
  "email": "test@example.com",
```

```
  "password": "1234"
```

```
}
```

HTTP 요청과 응답

HTTP 응답(Response)

- 서버가 클라이언트의 요청을 처리한 후 보내는 메시지
- 응답 구조
 - 상태 줄: 요청의 성공/실패를 알려주는 상태 코드
 - 요청 헤더: 응답 정보(형식, 길이 등)
 - 요청 바디: 실제 데이터(JSON, HTML 등)

HTTP 요청과 응답

HTTP 응답(Response)

- 서버가 클라이언트의 요청을 처리한 후 보내는 메시지

```
HTTP/1.1 201 Created          ← 상태 줄 (Status Line)
Content-Type: application/json; charset=utf-8    ← 응답 헤더 (응답 형식)
Location: https://jsonplaceholder.typicode.com/posts/101
Date: Tue, 16 Jul 2025 04:35:00 GMT
Content-Length: 78
```

```
{                                ← 응답 바디 (본문: JSON 데이터)
  "id": 101,
  "title": "Open API 실습 중입니다",
  "body": "이 글은 테스트용 게시물입니다.",
  "userId": 1
}
```

HTTP 요청과 응답

자주 사용하는 HTTP 상태 코드

코드	의미	설명
200	OK	요청 성공
201	Created	새 자원 생성 성공 (POST)
400	Bad Request	요청 형식 오류(JSON 누락, 형식 오류)
401	Unauthorized	인증 실패(토큰 누락)
403	Forbidden	권한 없음
404	Not Found	자원 없음
500	Internal Server Error	서버 오류(코드 오류)

HTTP 메소드 실습

JSONPlaceholder

- 실습, 테스트, 연습용으로 많이 쓰이는 가짜 REST API 사이트

JSONPlaceholder

[Guide](#) [Sponsor this project](#) [Blog](#) [My JSON Server](#)

{JSON} Placeholder

Free fake and reliable API for testing and prototyping.

Powered by [JSON Server](#) + [LowDB](#).

Serving ~3 billion requests each month.

<https://jsonplaceholder.typicode.com>

HTTP 메소드 실습

JSONPlaceholder

■ 예시 리소스 주소

- 게시물(Post) 전체 조회

<https://jsonplaceholder.typicode.com/posts>

- 게시물 ID 1번 조회

<https://jsonplaceholder.typicode.com/posts/1>

- 사용자 목록

<https://jsonplaceholder.typicode.com/users>

- 특정 사용자 ID

<https://jsonplaceholder.typicode.com/users/3>

- 댓글 조회

<https://jsonplaceholder.typicode.com/comments>

실습 파일

- <https://buly.kr/7QMQrJ6> 에 접속
- 사용 파일
 - 실습 파일: json.ipynb
 - JSON 파일: load.json

REST API

REST(Representational State Transfer, 표현 상태 전달)

- 웹에서 자원을 효과적으로 주고받기 위한 API 설계 방식
- 웹에서 데이터를 주고 받을 때의 약속
 - “자원은 URL로, 동작은 HTTP 메서드로 표현하자”
- 웹 서비스에서 가장 널리 사용되며, 구현이 간단하고 직관성이 높아 프론트엔드와 백엔드 통신에 적합

REST API

REST 등장 배경

- 과거에는 API 설계가 개발자, 플랫폼마다 모두 다름
 - 불명확한 요청 보낼 주소, 예측 불가능한 기능
- API를 누구나 쉽게 이해하고, 사용 가능하도록 만든 것이 REST

REST API

REST 구성 요소: 자원(Resource)

- 시스템 내에서 고유하게 식별 가능한 정보 단위
 - 서버에서 접근하거나 조작하려는 모든 대상
 - 게시물, 사용자, 상품, 주문, 댓글 등
 - 각 자원은 고유한 식별자(URI)를 가짐
 - /users, /products, /posts/1

REST API

URL과 URI의 차이점

- URL: Uniform Resource Locator
 - 인터넷 상 자원의 위치(주소)
 - 예: <https://example.com/index.html>
- URI: Uniform Resource Identifier
 - 인터넷 상의 자원을 식별하기 위한 문자열의 구성(식별자)
 - 예: <https://example.com>, <mailto:abc@example.com>, <urn:isbn:9783161484100>
- URI는 URL을 포함

REST API

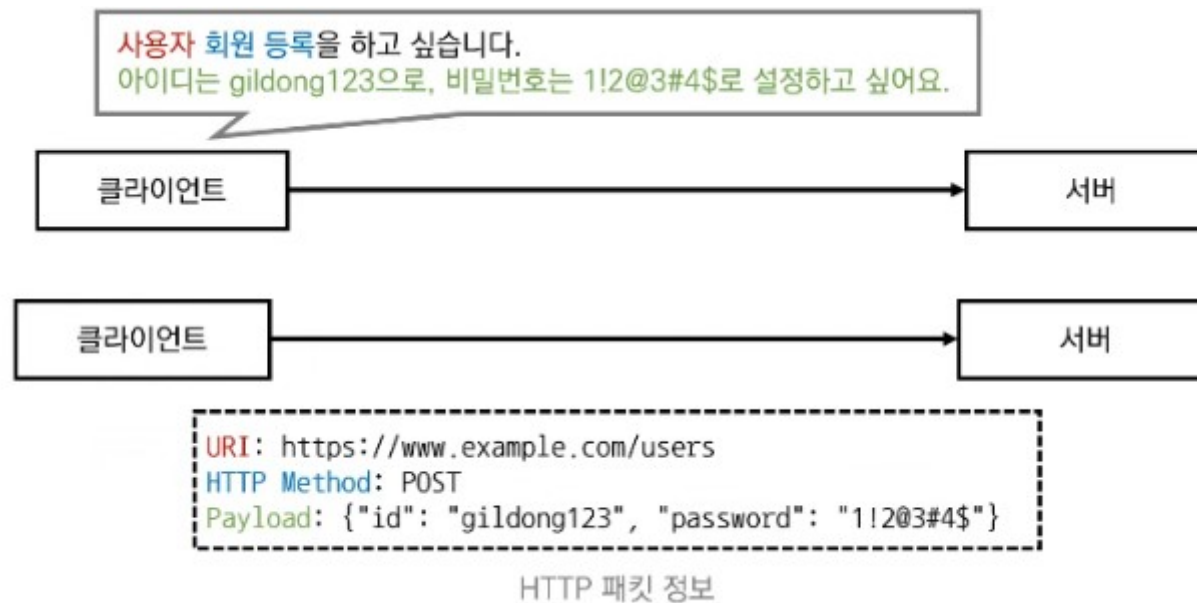
REST 구성 요소: 행위(Verb)

- HTTP 메서드
 - GET: Read
 - POST: Create
 - PUT/PATCH: Update
 - DELETE: Delete

REST API

REST 구성 요소: 표현(Representation)

- 클라이언트와 서버가 데이터를 주고받는 형식
 - 보통 JSON 또는 XML



REST API

REST의 설계 원칙

- URI는 자원을 나타내고 동작은 HTTP 메서드로 구분
 - 조회, 생성, 수정, 삭제 등 분리 설계
- 예시
 - GET /users → 사용자 목록 조회
 - POST /posts → 게시물 등록
 - DELETE /users/1 → ID 1번 사용자 삭제

REST API

REST의 특징

- 서버-클라이언트 구조(Loose Coupling)
 - 역할을 구분시켜 서로 간의 의존성을 줄임
- 무상태(Stateless)
 - 클라이언트의 Context(세션, 쿠키 등)을 서버에 저장하지 않음
 - 서버는 각 요청을 별개의 것으로 인식해 처리(이전 요청이 다음 요청에 연관 X) → 서버와 클라이언트가 독립적으로 확장 가능

REST API

REST의 특징

- 상태 전이 용이성(Stateless Interactions)
 - 각 요청이 독립적이므로 서버는 별도의 세션 관리 필요 X
 - 서버 구현의 단순화
- 캐시 처리 가능(Cacheable)
 - HTTP 프로토콜의 캐싱 기능을 그대로 사용 가능
 - 서버 부하를 줄이고 성능 향상

REST API

REST의 특징

- 계층 구조(Layered System)
 - 서버 시스템이 여러 단계(계층)로 구성되어 있어 클라이언트는 서버를 구분하지 않고 요청을 보낼 수 있음
 - 중간에 계층을 자유롭게 추가 가능
- 인터페이스 일관성
 - HTTP 표준 프로토콜을 따르는 모든 플랫폼에서 사용 가능
- 자기 설명성(Self-Descriptiveness)
 - 요청 메시지만 보고도 쉽게 이해 가능한 구조

REST API

REST의 장점

- 일관성
 - URI만 보고 기능 유추 가능
- 표준 HTTP 사용
 - 다양한 시스템 간의 통신 방식 통일
 - 단순하고 예측 가능한 방식으로 통신 가능
- 프론트엔드와 백엔드의 분리가 쉬움
 - 팀 간 협업 효율 증가, 유지 보수 용이
- 확장성과 재사용성이 높음

REST API

REST의 단점

- 복잡한 데이터 요청에는 비효율적
 - 너무 단순한 구조 → GraphQL(API 쿼리 언어)이 보완
- 오버패칭/언더패칭 발생
 - 필요한 데이터만 가져오기 어려움
- 엄격한 규칙 필요
 - 적절한 URI 설계 필요
- 보안성 문제
 - Stateless 구조를 가지고 있으므로 보안 문제 발생
 - HTTPS 같은 보안 프로토콜 사용 또는 인증 기능 추가

REST API

REST의 단점

- 테스트가 어려움
 - 다양한 클라이언트와 서버 간에 상호작용하는 기술이므로 테스트가 어려움
- 복잡한 API 설계는 유지 보수가 어려움
 - 여러 개의 자원, 각 자원마다의 HTTP 메소드가 있으므로 유지 보수가 어려움
- 표준이 없음
 - 아키텍처 스타일이므로 기업, 플랫폼마다 자유롭게 구현 가능

REST API

REST API

- REST 원칙을 적용한 HTTP 기반 API
- Open API, 마이크로 서비스 등을 제공하는 기업 대부분은 REST API 제공

REST API

REST API의 특징

- 각 요청이 어떤 동작이나 정보를 위한 것인지, 요청 자체로 추론 가능
- 확장성과 재사용성 → 유지보수와 운용이 편리
 - 자원 중심 설계와 HTTP 프로토콜
 - 독립적인 서비스 단위
 - 자기 설명적인 메시지

REST API

REST API의 디자인 가이드

- URI는 정보의 자원 표현
- 자원에 대한 행위는 HTTP 메소드로 표현

REST API

REST API 설계 원칙

- URI를 작성할 때는 명사 사용(예: /users, /posts)
- URI에 작성되는 영어는 복수형으로 작성
- URI를 작성할 때는 소문자 사용
- URL의 마지막에는 슬래시(/)를 포함하지 않음
- 계층 관계를 나타낼 때는 슬래시 구분자 사용
 - URI 경로 부분에서 자원 간의 계층적 관계를 나타내기 위해

REST API

REST API 설계 원칙

- 언더바(_) 대신 하이픈(-)
 - 가독성을 위해 긴 path를 표현하는 단어는 하이픈 사용
- 행위는 HTTP 메서드를 사용해 전달
- HTTP 응답 상태 코드 활용
- 파일 확장자는 URI에 포함하지 않음
- 조회 시 쿼리 활용(예: /users?page=2&count=10)
- 리소스 간 연관이 있다면 '/리소스명/리소스ID/관계가 있는 다른 리소스명'으로 표현

REST API

REST API의 설계 예시

동작	URL(자원 구조)	메서드	예시
사용자 전체 조회	/users	GET	GET /users
사용자 추가	/users	POST	POST /users
사용자 조회	/users/5	GET	GET /users/5
사용자 수정	/users/5	PUT	PUT /users/5
사용자 삭제	/users/5	DELETE	DELETE /users/5

RESTful API

개념

- REST 원칙을 철저히 지킨 API
- RESTful한 API는 요청을 보내는 주소만으로도 어떤 것을 요청하는지 파악 가능

공공데이터와 Open API

- 공공 데이터
- Open API
- Open API 발급

공공 데이터

소개

- 정부, 공공기관, 지자체 등이 보유한 데이터를 누구나 자유롭게 사용할 수 있도록 공개한 데이터
- 공공데이터포털(data.go.kr)
 - 기상청, 환경부, 국토교통부 등의 여러 기관이 Open API로 데이터 제공

Open API

소개

- 공공 데이터 또는 민간 데이터를 API 형태로 제공해 프로그래밍으로 데이터를 자동으로 가져올 수 있게 만든 인터페이스
- HTTP 요청으로 전송하는 JSON, XML 형식
- 데이터 분석, 서비스 개발 등에 유용
- 장점
 - 자동화, 실시간성, 연동성, 확장성

Open API

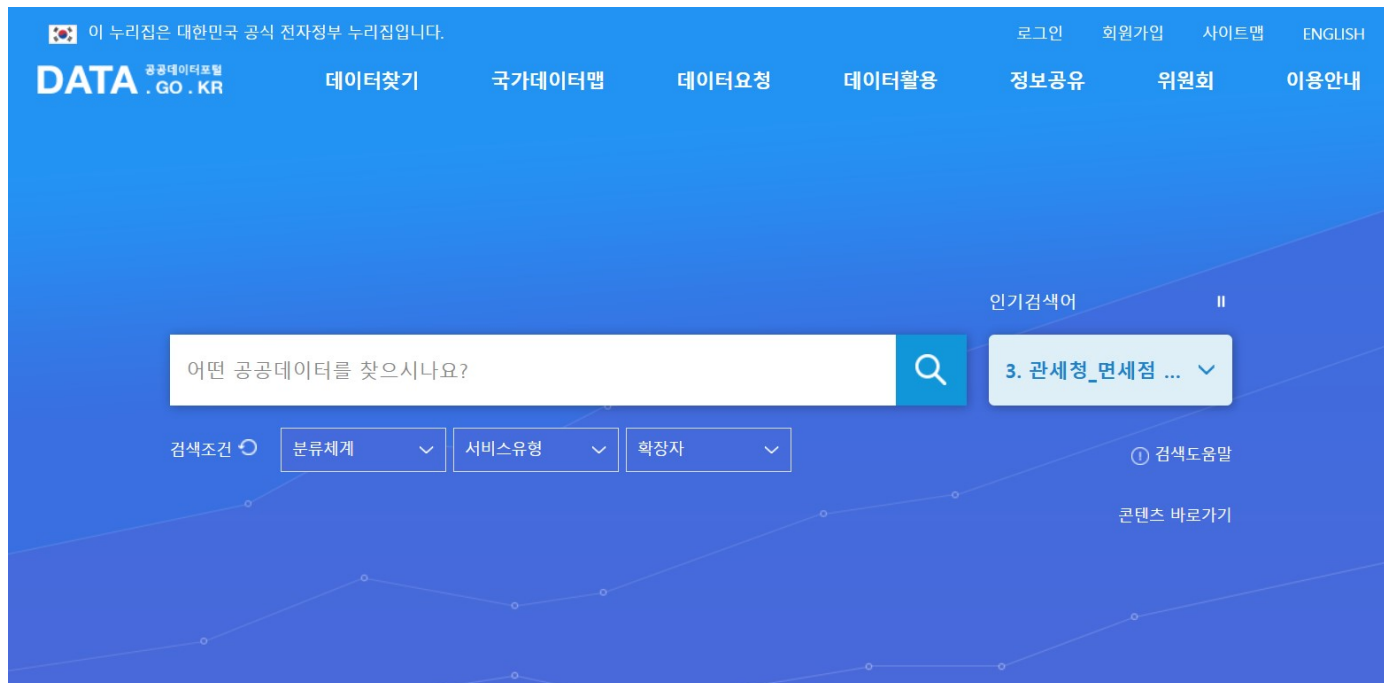
활용 예시

- 날씨 데이터
 - 날씨 기반 알림 앱, 농작물 수분 조절 시스템 등
- 대중교통 도착 정보
 - 정류장 디지털 전광판, 출근 알림 챗봇, 지도 앱
- 부동산 실거래가 정보
 - 부동산 가격 변화 추이 그래프, 예측 모델 학습

날씨 정보 가져오기

공공데이터포털 회원 가입

- 공공데이터 포털 접속 <https://www.data.go.kr>



날씨 정보 가져오기

공공데이터포털 회원 가입

- 메인 페이지에서 우측 상단 **회원가입** 클릭
- 회원가입 절차
 - 가입 유형 선택: **개인회원** 선택
 - 이용약관 전체 동의 → 다음 단계로 이동
 - 이름, 이메일, 비밀번호 등 기본 정보 입력
 - 이메일 인증 절차 진행

날씨 정보 가져오기

기상청 단기예보 조회서비스 활용 신청

■ 검색 창에 기상청 단기예보 조회서비스 검색

The screenshot shows the top navigation bar of the DATA.GOV.KR portal. The header includes the text '이 누리집은 대한민국 공식 전자정부 누리집입니다.' and a language selector with 'ENGLISH'. The main navigation menu contains links for '데이터찾기', '국가데이터맵', '데이터요청', '데이터활용', '정보공유', '위원회', and '이용안내'. The search bar is prominently displayed with the text '기상청 단기예보 조회서비스' entered. Below the search bar are filters for '검색조건', '분류체계', '서비스유형', and '확장자'. To the right of the search bar, there is a section for '인기검색어' (Popular Search Terms) and a link to '검색도움말' (Search Help).

이 누리집은 대한민국 공식 전자정부 누리집입니다.

로그아웃 마이페이지 사이트맵 ENGLISH

DATA 공공데이터포털 .GOV.KR 데이터찾기 국가데이터맵 데이터요청 데이터활용 정보공유 위원회 이용안내

인기검색어

기상청 단기예보 조회서비스

검색조건 분류체계 서비스유형 확장자

① 검색도움말

콘텐츠 바로가기

날씨 정보 가져오기

기상청 단기예보 조회서비스 활용 신청

- 오픈 API 목록 → “기상청_단기예보 ((구)_동네예보) 조회서비스” → 우측하단의 “활용신청” 선택

오픈 API (24건)

더보기 >

과학기술

국가행정기관

미리보기

XML

JSON

기상청_단기예보 ((구)_동네예보) 조회서비스

초단기실황, 초단기예보, 단기((구)동네)예보, 예보버전 정보를 조회하는 서비스입니다.

제공기관 기상청 수정일 2024-12-20 조회수 505781 활용신청 42985 키워드 단기예보,초단기실황,초단기예보

활용신청

날씨 정보 가져오기

기상청 단기예보 조회서비스 활용 신청

- 활용 목적: 웹 사이트 개발 선택
 - 공공API 활용에 대한 실습 교육 입력

활용목적 선택

*표시는 필수 입력항목입니다.

*활용목적

☒ 웹 사이트 개발 ☐ 앱개발 (모바일,솔루션등) ☐ 기타 ☐ 참고자료 ☐ 연구(논문 등)

공공API 활용에 대한 실습 교육

날씨 정보 가져오기

기상청 단기예보 조회서비스 활용 신청

- 이용허락범위: 동의합니다 클릭
- 활용신청 버튼 클릭

라이선스 표시

* 이용허락범위

저작자표시

☒ 동의합니다.

취소

활용신청

날씨 정보 가져오기

인증키 발급 확인 위치

- 상단 메뉴 → 마이페이지 클릭
- 왼쪽 메뉴: 데이터 활용 → OpenAPI → “인증키 발급 현황”

마이페이지

데이터 활용

Open API

활용신청 현황

인증키 발급현황

파일 데이터

관심 데이터

인증키 발급현황

총 1건

구분	발급일자	재발급여부	인증키
일반	2021/09/07	신규발급	XU%2FfC%2FPP36irB8owS%2BdGDPrKch7TBFxBDW25F8eMkpM%2FoC4

일반 인증키 재발급

날씨 정보 가져오기

신청현황에서 인증키 확인 방법

- 신청한 API 목록 중 “기상청_단기예보 조회서비스” 항목 선택
- 서비스 정보: 일반 인증키(Decoding) 복사 후 저장

서비스정보

참고문서	기상청41_단기예보 조회서비스_오픈API활용가이드_241128.zip
데이터포맷	JSON+XML
End Point	https://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 사용하시: * 향후 포털에서 더 명확한 정보를 제공하기 위해 노력하겠습니다.	
일반 인증키 (Encoding)	XU%2FfC%2FPP36irB8owS%2BdGDPrKch7TBFxBDW25F8eMkpM%2FoC4G5Ubm51Vbl44Up8hFC937oUS72RVjdhWO54
일반 인증키 (Decoding)	XU/fC/PP36irB8owS+dGDPrKch7TBFxBDW25F8eMkpM/oC4G5I

날씨 정보 가져오기

참고 사항

- 인증키는 URL 요청 시 반드시 포함되어야 함
- 키 유출 주의(외부 공개 금지)
- 하나의 키로 여러 API 호출 가능
- 일일 제한량

날씨 정보 가져오기

상세 API 기능 설명

■ Call Back URL 복사

상세기능 번호	1	상세기능 유형	조회 (목록)
상세기능명(국문)	초단기실황조회		
상세기능 설명	실황정보를 조회하기 위해 발표일자, 발표시각, 예보지점 X 좌표, 예보지점 Y 좌표의 조회 조건으로 자료구분코드, 실황값, 발표일자, 발표시각, 예보지점 X 좌표, 예보지점 Y 좌표의 정보를 조회하는 기능		
Call Back URL	http://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getUltraSrtNcst		
최대 메시지 사이즈	[1764] byte		
평균 응답 시간	[100] ms	초당 최대 트래잭션	[30] tps

개발계정 상세보기 페이지에 있는 "기상청41_단기예보 조회서비스_오픈API활용가이드_241128.zip"파일

날씨 정보 가져오기

요청 메시지 명세

항목명(영문)	항목명(국문)	항목크기	항목구분	샘플데이터	항목설명
serviceKey	인증키	100	1	인증키 (URL Encode)	공공데이터포털에서 발급받은 인증키
numOfRows	한 페이지 결과 수	4	1	10	한 페이지 결과 수 Default: 10
pageNo	페이지 번호	4	1	1	페이지 번호 Default: 1
dataType	응답자료형식	4	0	XML	요청자료형식(XML/JSON) Default: XML
base_date	발표일자	8	1	20210628	'21년 6월 28일 발표
base_time	발표시각	4	1	0600	06시 발표(정시단위) -매시각 10분 이후 호출
nx	예보지점 X 좌표	2	1	55	예보지점의 X 좌표값 *별첨 엑셀 자료 참조
ny	예보지점 Y 좌표	2	1	127	예보지점의 Y 좌표값 *별첨 엑셀 자료 참조

※ 항목구분 : 필수(1), 옵션(0), 1 건 이상 복수건(1..n), 0 건 또는 복수건(0..n)

개발계정 상세보기 페이지에 있는 "기상청41_단기예보 조회서비스_오픈API활용가이드_241128.zip"파일

날씨 정보 가져오기

오픈 API 활용 가이드 살펴보기

- '기상청41_단기예보
조회서비스_오픈API활용가이드_241128.zip' 다운로드
- 기상청41_단기예보 조회서비스_오픈API활용가이드.doc 열기
- p. 4: 나. 상세 기능 목록, 다. 상세기능내역
- p. 7: # 코드값 정보

날씨 정보 가져오기

HTTP GET 요청 주소 및 파라미터 추출 방법

- 활용 신청한 상세 기능 정보: "초단기 실황조회"
- 미리보기 [확인] 선택

활용신청 상세기능정보

NO	상세기능	설명	일일 트래픽	미리보기
1	초단기실황조회	실황정보를 조회하기 위해 발표일자, 발표시각, 예 보지점 X 좌표, 예보지점 Y 좌표의 조회 조건으로 자료구분코드, 실황값, 발표일자, 발표시각, 예보지 점 X 좌표, 예보지점 Y 좌표의 정보를 조회하는 기 능	10000	확인

날씨 정보 가져오기

HTTP GET 요청 주소 및 파라미터 추출 방법

- 요청 변수 항목에서 아래 항목 수정
 - dataType : JSON / base_date : 20250716
- **미리보기** 선택

요청변수(Request Parameter) 닫기

항목명	샘플데이터	설명
ServiceKey	-	공공데이터포털에서 받은 인증키
pageNo	1	페이지번호
numOfRows	1000	한 페이지 결과 수
dataType	JSON	요청자료형식(XML/JSON) Default: XML
base_date	20250716	'21년 6월 28일 발표
base_time	0600	06시 발표(정시단위)
nx	55	예보지점의 X 좌표값
ny	127	예보지점의 Y 좌표값

미리보기

날씨 정보 가져오기

HTTP GET 요청 주소 및 파라미터 추출 방법

- 표시되는 미리보기 사이트의 URL 주소 모두 복사



Postman

소개

- REST API 요청을 쉽게 테스트할 수 있도록 도와주는 툴
- 웹 브라우저가 아닌 별도의 도구에서 요청을 구성하고 직접 응답을 받아 확인 가능

Postman

장점

- 코딩 없이 API 테스트 가능
 - API 주소, 파라미터, 헤더만 입력하면 바로 결과 확인 가능
- 요청/응답 구조 확인이 쉬움
 - JSON 응답 구조를 보기 좋게 표시
- 빠르게 반복 테스트 가능
 - 파라미터를 바꾸고 바로 재요청, 복사, 저장 가능
- 에러 디버깅
 - 응답 코드, 메시지, 헤더를 바로 확인 가능

Postman

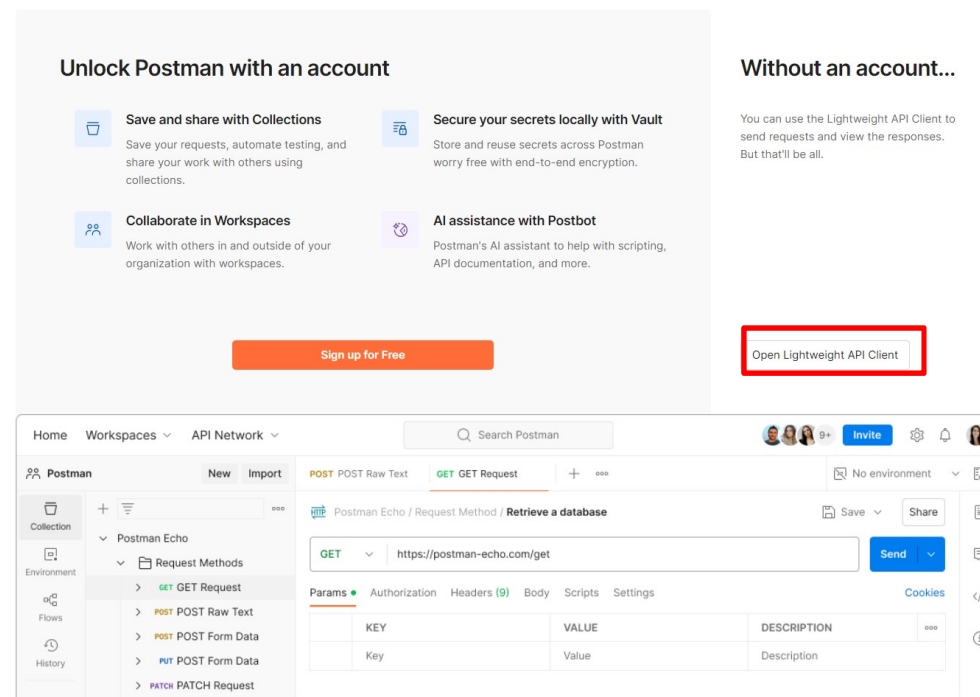
설치

- <https://www.postman.com/downloads/>
- Windows / macOS 버전 모두 제공
- 설치 후 로그인 없이 바로 사용 가능(게스트 모드)
 - **Continue without an account** 클릭

Postman

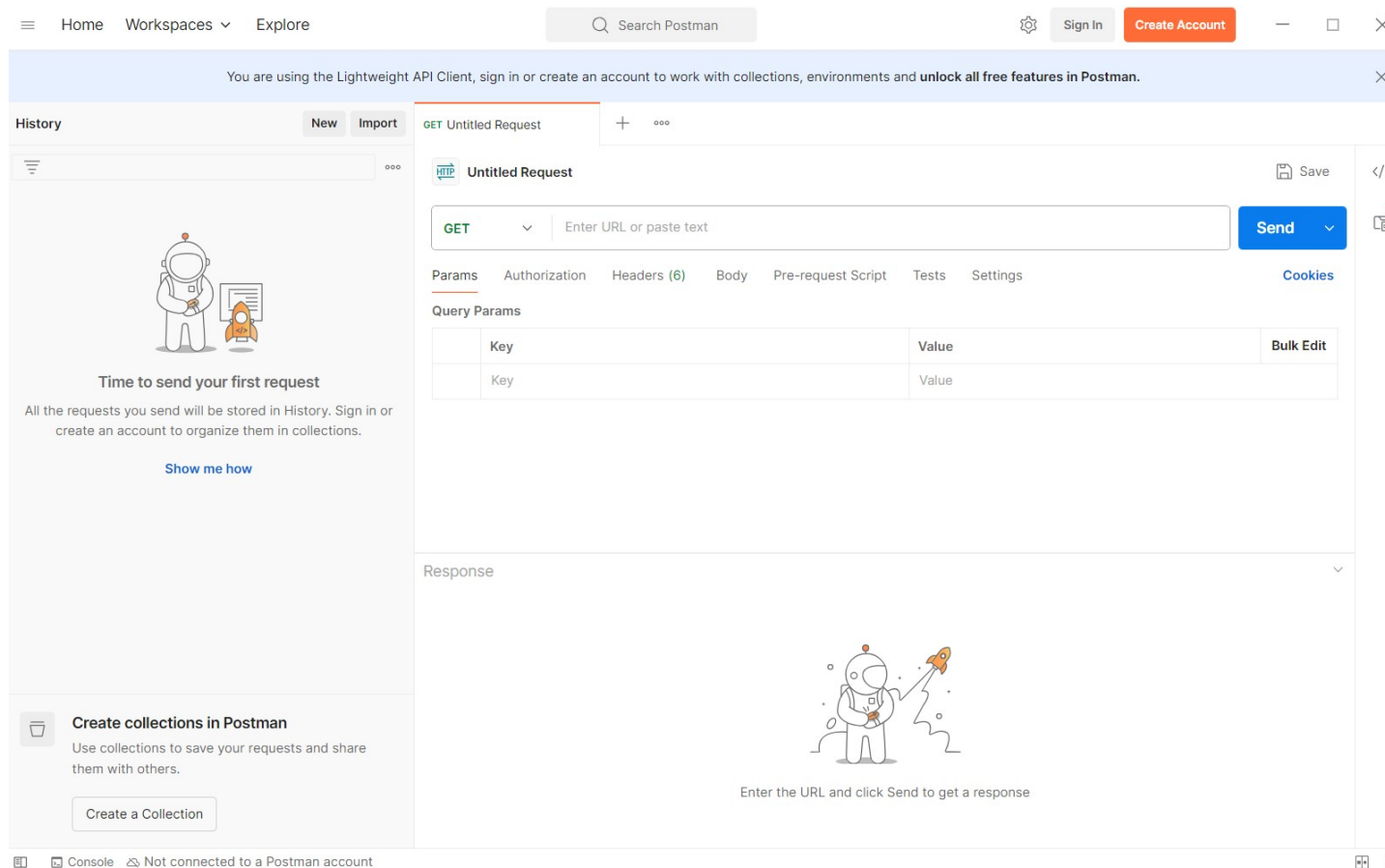
설치

■ Open Lightweight API Client 클릭



Postman

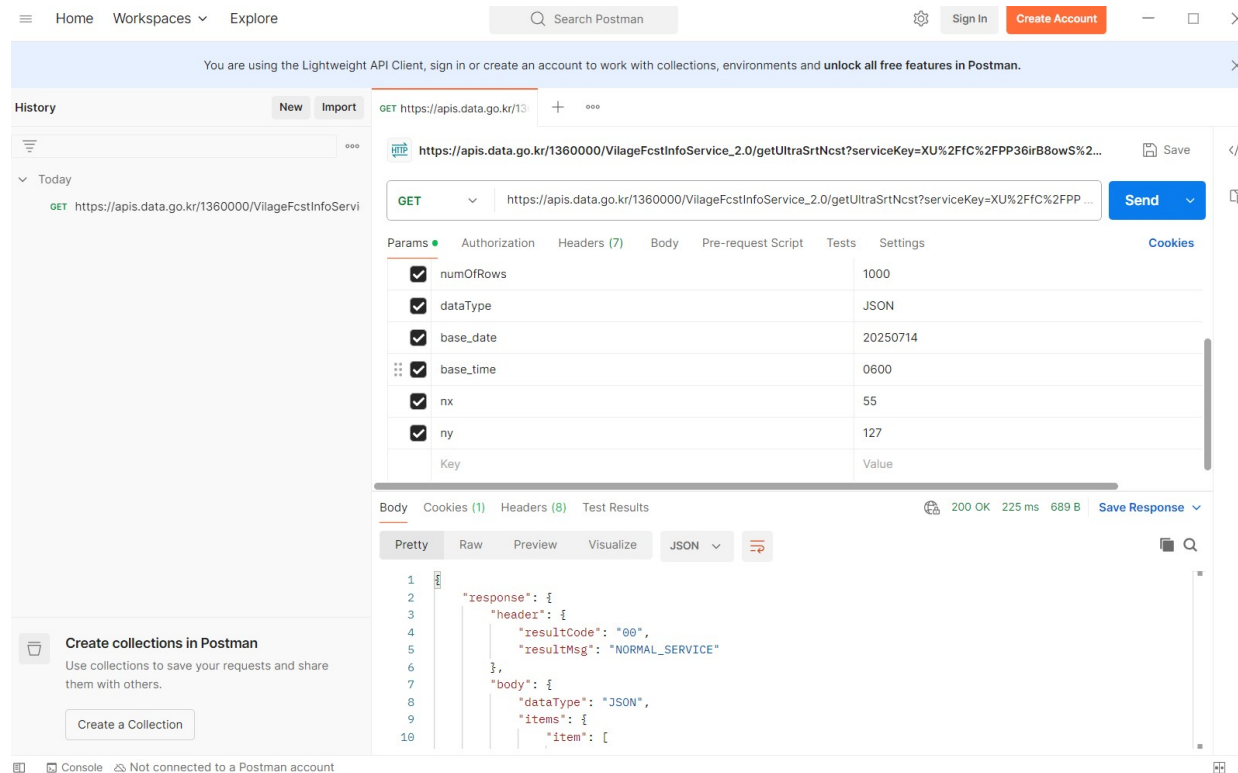
실행 화면



Postman

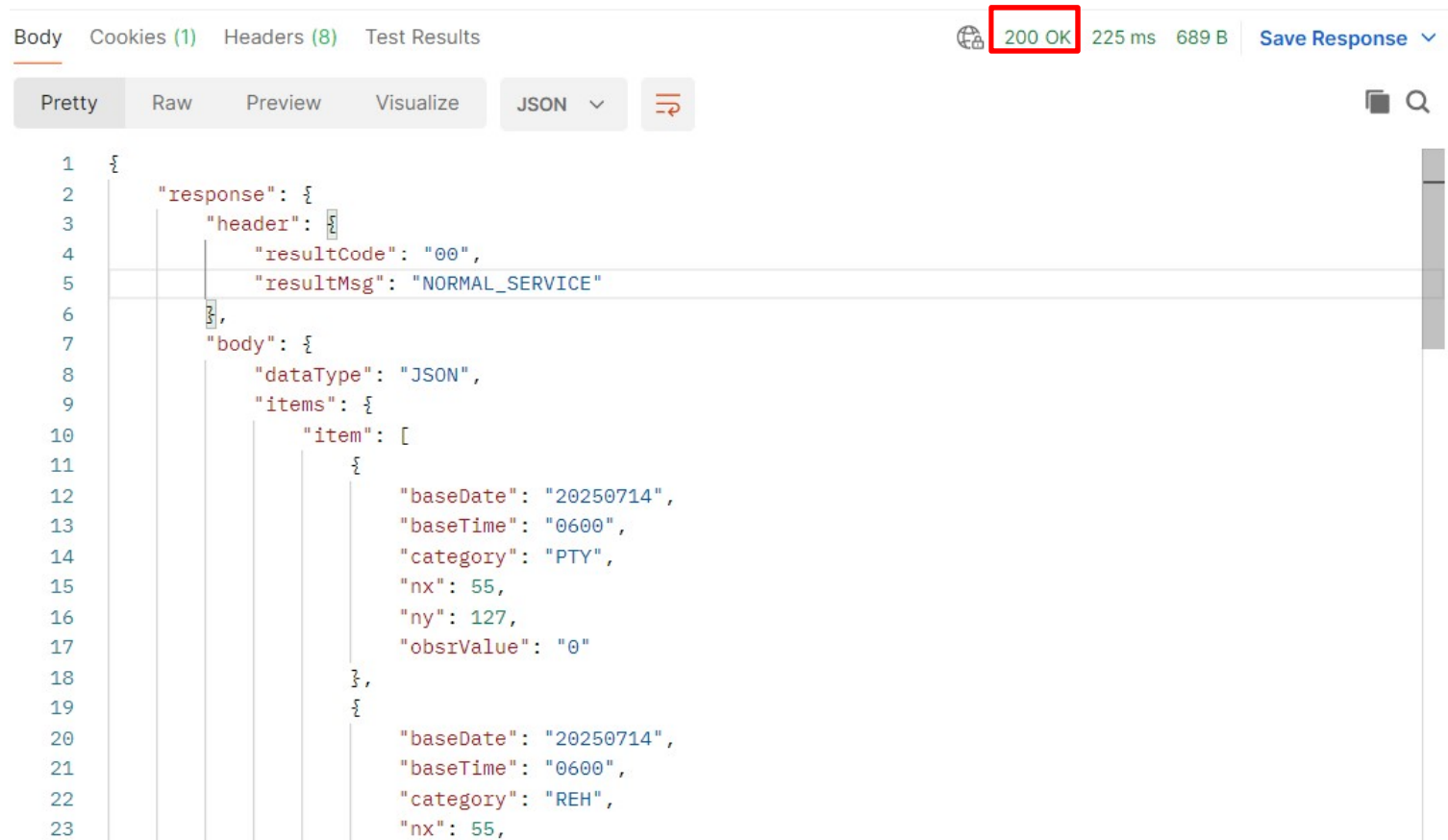
GET 실습

- 복사한 전체 URL을 Postman의 GET 주소에 붙여 넣기



Postman

API 응답 확인



A screenshot of the Postman application showing an API response. The top status bar indicates a 200 OK status, 225 ms response time, and 689 B body size. The response is displayed in the 'Body' tab, formatted as JSON. The JSON structure includes a 'response' object with 'header' and 'body' fields. The 'header' field contains 'resultCode' and 'resultMsg'. The 'body' field contains a 'data' object with 'items' and 'item' fields. The 'items' field contains an array of two objects, each with 'baseDate', 'baseTime', 'category', 'nx', 'ny', and 'obsrValue' fields.

```
1 {
2   "response": {
3     "header": {
4       "resultCode": "00",
5       "resultMsg": "NORMAL_SERVICE"
6     },
7     "body": {
8       "dataType": "JSON",
9       "items": {
10        "item": [
11          {
12            "baseDate": "20250714",
13            "baseTime": "0600",
14            "category": "PTY",
15            "nx": 55,
16            "ny": 127,
17            "obsrValue": "0"
18          },
19          {
20            "baseDate": "20250714",
21            "baseTime": "0600",
22            "category": "REH",
23            "nx": 55,
```

Postman

응답 구조 해석

- 응답 형식: JSON
- 최상위 구조: response → body → items → item
- 각 item은 하나의 기상 요소(category) 정보 포함
- 예: {
 "category": "T1H", // 기온
 "obsrValue": "22.3" // 측정값
}

JSON의 구조

- JSON이란?
- JSON 구조

JSON이란?

JSON(JavaScript Object Notation)

- 구조화된 데이터를 표현하기 위한 문자 기반의 표준 포맷
- 일반적으로 웹 어플리케이션에서 데이터를 전송할 때 사용하는 데이터 교환의 표준 형식

JSON이란?

JSON의 장점

- 가독성
 - 텍스트 형식을 사용해 인간과 컴퓨터가 데이터를 빠르게 파악 가능
- 호환성
 - 다양한 프로그래밍 언어와 플랫폼에서 데이터 교환 가능
- 텍스트 기반 형식이라 가벼움

JSON이란?

JSON의 장점

- 유연성
 - 다양한 정보를 하나의 컬럼으로 적재 가능
 - TTS 전사 데이터, 크롤링 결과, 로그 이벤트, 설정 파일 등 어떤 데이터를 다룰 때도 잘 맞는 구조
 - 형식이 고정되지 않은 반정형 데이터 저장 가능
- 파싱이 쉬운 데이터 형식

JSON이란?

활용 사례

- 웹 API와 데이터 교환
- 설정 파일 저장
 - 간단하고 사용하기 쉬운 데이터 형식
- 구성 데이터 저장
 - 복잡한 데이터 구조
- 데이터 저장 및 전송

JSON이란?

JSON과 XML 비교

- 웹에 데이터를 저장하고 전송할 수 있는 형식
- JSON
 - 웹에서 데이터를 교환할 때 사용(API)
 - 간단하고 사용하기 쉬운 데이터 형식
- XML(Extensible Markup Language)
 - 태그 기반 마크업 언어
 - 구조화된 데이터를 표준 형식으로 교환할 때 사용(구 버전 API, 문서 기반 시스템)
 - 복잡한 데이터 구조(많은 태그)

JSON이란?

JSON과 XML 비교

JSON

```
{  
  "person": {  
    "name": "Alice",  
    "age": 30,  
    "email": "alice@example.com"  
  }  
}
```

XML

```
<person>  
  <name>Alice</name>  
  <age>30</age>  
  
  <email>alice@example.  
com</email>  
</person>
```

JSON 구조

파일 형식

- 데이터는 키-값 쌍으로 표현
 - 키: 문자열(str)
 - 값: 문자열(str), 숫자(number), 불리언(true, false), 객체(object), 배열(array), null
- 문자열은 "(큰 따옴표)만 가능
- 'json', 'jsonl' 파일 형식으로 저장

JSON 구조

Python 객체와 JSON 표현의 자료형 매핑 관계

Python 객체	JSON 표현
딕셔너리(dict)	객체(object)
List, tuple	배열(array)
문자열(str)	문자열(string)
Int, float	숫자(number)
True, False	True, false
None	null
코드 안에서 직접 사용	네트워크/파일 전송
키, 문자열에 ‘ 또는 “ 사용 가능	반드시 “ 사용

JSON 구조

Python 객체와 JSON 표현의 자료형 매핑 관계

```
# Python
data = {
    'name': 'Alice',
    'age': 30,
    'is_active': True,
    'score': None
}
```

```
# JSON
{
    "name": "Alice",
    "age": 30,
    "is_active": true,
    "score": null
}
```

JSON 구조

객체와 배열

- 객체(object)
 - 중괄호 {}로 표현
 - 키-값 쌍을 쉼표로 구분
- 배열(Array)
 - 대괄호 []로 표현, 순서 있는 값들의 목록
 - 값을 쉼표로 구분

JSON 구조

JSON 객체/배열 예시

객체

```
{  
  "name": "John",  
  "age": 30,  
  "isStudent": true  
}
```

배열

```
[  
  "blue",  
  "red",  
  "yellow"  
]
```

JSON 구조

중첩 계층 구조

- 객체 안에 다른 객체 또는 배열을 중첩해 사용 가능

```
{  
  "person": {                # 키: person  
    "name": "Anne",  
    "age": 32  
  },  
  "hobbies": ["reading", "hiking"] # 키: hobbies  
}
```

JSON 파싱 및 데이터 추출

- JSON 파싱 및 데이터 추출(실습)

JSON 파싱

파싱(Parsing)

- 데이터를 컴퓨터가 이해하고 처리할 수 있는 구조화된 형태로 변환하는 과정
- 데이터 분석 → 의미 있는 구성 요소로 분해 → 정보 추출

JSON 파싱

json 모듈

- Python 내장 모듈
- JSON 데이터를 파싱하고 생성하는 모듈
 - import json

JSON 파싱

Request 라이브러리

- 파이썬에서 HTTP 요청을 보내고 받는 데 사용되는 라이브러리
 - import requests

실습 파일

- <https://buly.kr/7QMQrJ6> 에 접속
- 사용 파일
 - 실습 파일: json.ipynb
 - JSON 파일: load.json

기상 API 데이터 전처리 및 저장

- SQLite란?
- API 데이터 전처리 및 DB 저장(실습)

SQLite란?

개념

- 가볍고 빠른 파일 기반의 관계형 데이터베이스
- 내장형 DB라 별도의 서버 설치가 필요 없음
 - 로컬 컴퓨터에 '.db' 파일로 저장
- SQL 표준을 대부분 지원
- 다른 DB로 마이그레이션 가능
- 다중 사용자의 동시 접속 제한
 - 동시에 읽기는 가능, 쓰기는 불안정

SQLite란?

Python에서의 사용 예시

```
import sqlite3 # 내장 Sqlite 모듈 호출

conn = sqlite3.connect("weather.db") # DB 연결, 없으면 새로 생성
cursor = conn.cursor() # SQL 명령어를 실행할 수 있는 cursor 객체 생성

cursor.execute("""
CREATE TABLE IF NOT EXISTS weather (
    date TEXT,
    temperature REAL
)
""")
# weather 테이블에 한 행의 데이터 삽입, ?는 자리 표시자
cursor.execute("INSERT INTO weather VALUES (?, ?)", ("2025-07-15", 30.2))
conn.commit() # 변경 사항을 DB에 반영
conn.close() # DB 연결 종료
```

SQLite란?

Python에서의 사용 예시: with 문

- 여러 쿼리를 연속으로 실행할 때, 하나로 묶어 사용

```
import sqlite3

with sqlite3.connect("weather.db") as conn:
    cursor = conn.cursor()
    cursor.execute("SELECT * FROM weather")
    rows = cursor.fetchall() # SQL 쿼리 실행 결과를 모두 가져옴
    print(rows)
# with 블록이 끝나면 conn.close() 자동 호출됨
```


SQLite란?

SQLAlchemy

- Python 코드로 SQL을 작성하고, 객체 지향 방식으로 데이터베이스를 제어할 수 있게 해주는 라이브러리
- SQL 문을 직접 사용하지 않고 객체처럼 다룰 수 있어 안정하고 간결하게 표현
- 다양한 DB 지원
- 유지보수가 쉬움
- SQL 인젝션 방지 자동 처리

SQLite란?

SQLAlchemy 사용 예시

```
i# 데이터 타입 지정
from sqlalchemy import create_engine, String, Float, Integer, DateTime

# SQLAlchemy 엔진 생성 (to_sql의 dtype 인자 활용)
engine = create_engine('sqlite:///weather_data.db', echo=False) # echo=False: 실행되는 SQL 쿼리를
콘솔에 출력 x

# dtype 매핑
dtype_map = {
    "datetime": DateTime(),
    "category": String(10), # 코드
    "category_name": String(50), # 한글
    "obsrValue": Float(),
    "nx": Integer(),
    "ny": Integer()
}

# 저장(dtype 지정): weather_data DB 내 테이블로 저장
df.to_sql("observed_weather", con=engine, if_exists="append", index=False, dtype=dtype_map)
```

SQLite란?

SQLAlchemy

- 타입 명시가 정확하고, 내부적으로 더 안정적인 SQL 코드 생성
- 확장성, 유지보수, 복잡한 테이블 구조에 유리
- 엔진
 - DB 연결 설정
 - 연결 유지 관리
 - ORM(Object Relational Mapping) 클래스나 raw SQL 실행

실습 파일

- <https://buly.kr/7QMQRJ6> 에 접속
- 사용 파일
 - weather_api_training.ipynb

실시간 기상 API 데이터 전처리 및 저장 자동화

- 자동화 순서
- 실시간 기상 API 데이터 전처리 및 저장 자동화(실습)
- 마무리

자동화 순서

1. 외부(Open API)에서 데이터를 요청 → 응답(JSON) 받음
2. JSON 포맷을 Python 딕셔너리로 변환
3. 추출된 데이터를 DataFrame 형식으로 변환
4. 데이터 전처리
5. 전처리된 데이터를 SQLite DB에 테이블로 저장 (.db 파일)
6. 주기적으로 반복 실행 (while + time.sleep())

실습 파일

- <https://buly.kr/7QMQrJ6> 에 접속
- 사용 파일
 - 자동화 파이프라인: `weather_auto_fetcher.py`
 - 테스트: `auto_data.ipynb`

실습

- 자동화 파이프라인을 직접 완성해 보세요.
 - 정상적으로 실행되면 아래 메시지가 출력됩니다.
[2025-07-15 15:00:12.345678] 저장 완료: 8건
- 테스트 파일을 열어 결과를 확인해 보세요.

마무리

- API, REST, REST API, RESTful API
- JSON 형식
- JSON 파싱
- SQLite에 데이터 저장
- 자동화 파이프라인

Q&A

