빅카인즈 api

▼ 뉴스 검색 api

▼ code

```
import requests
import os
from dotenv import load_dotenv
from pprint import pprint

# 1. 환경 변수 로딩
load_dotenv()
API_KEY = os.getenv("BIGKINDS_KEY")

# 2. 요청 URL
url = "https://tools.kinds.or.kr/search/news"

# 3. 다양한 테스트 케이스 입력
search_conditions = {
    "query": "인공지능 OR AI OR 챗GPT",   # 🔍 키워드 실험 가능
    "from": "2025-04-01",                # 🗓️ 기간 조절 가능
    "until": "2025-04-15",
    "provider": ["서울경제"],            # 📰 너희 회사로 제한 (기타 안됨)
    "category": [],                    # 예: ["IT_과학"], ["경제"]
    "fields": ["title", "published_at", "content", "provider", "byline"]  # 어떤 데
}

# 4. 요청 payload 구성
payload = {
    "access_key": API_KEY,
    "argument": {
        "query": search_conditions["query"],
        "published_at": {
            "from": search_conditions["from"],
            "until": search_conditions["until"]
        },
        "provider": search_conditions["provider"],
        "category": search_conditions["category"],
        "category_incident": [],
```

```
        "provider_subject": [],
        "subject_info": [],
        "sort": {"date": "desc"},
        "return_from": 0,
        "return_size": 5,
        "fields": search_conditions["fields"]
    }
}

# 5. 요청 보내기
response = requests.post(url, json=payload)

# 6. 결과 출력
if response.status_code == 200:
    print("✅ 뉴스 검색 API 호출 성공!")
    result = response.json()
    documents = result["return_object"]["documents"]
    total_hits = result["return_object"]["total_hits"]

    print(f"\n📊 총 검색 결과: {total_hits}건\n")

    for i, doc in enumerate(documents, 1):
        print(f"----- {i} -----")
        for field in search_conditions["fields"]:
            print(f"{field}: {doc.get(field, '[없음]')}")
        print()
else:
    print("❌ 오류 발생:", response.status_code)
    print(response.text)
```

▼ 항목 정리

✅ [정리] 뉴스 검색 API (`/search/news`) 요청 항목 목록

항목명 (`argument` 기준)	필수 여부	설명	예시
`query`	✅	검색 키워드 / 조건	`"인공지능 OR AI""반도체 AND 투자"`
`published_at` → `from`	✅	시작 날짜 (YYYY-MM-DD)	`"2024-01-01"`

`published_at` → `until`	✅	종료 날짜 (해당일 포함)	`"2024-12-31"`
`provider`	⭕	언론사 이름 리스트	`["서울경제"]` *(다른 건 제한됨)*
`category`	⭕	기사 카테고리 (대/중 분류)	`["경제"]`, `["IT_과학"]`
`category_incident`	⭕	사건/사고 분류	`["범죄", "재해>자연재해"]`
`provider_subject`	⭕	언론사 주제 분류	`["경제", "부동산"]`
`subject_info` ~ `subject_info4`	⭕	뉴스ML 기반 주제 분류	`["정치", "국제"]`
`sort`	⭕	정렬 방식	`{"date": "desc"}{"title": "asc"}`
`return_from`	⭕	결과 시작 인덱스	`0`, `10`, `20`
`return_size`	⭕	결과 개수	`5`, `20`, `100` (최대 10,000)
`fields`	⭕	출력할 항목들 리스트	`["title", "content", "published_at"]`

🧾 예시 요청 바디 구성 샘플

```
{
  "access_key": "your_api_key_here",
  "argument": {
    "query": "인공지능 OR AI",
    "published_at": {
      "from": "2025-04-01",
      "until": "2025-04-15"
    },
    "provider": ["서울경제"],
    "category": ["IT_과학"],
    "sort": { "date": "desc" },
    "return_from": 0,
    "return_size": 5,
    "fields": ["title", "published_at", "byline"]
  }
}
```

▼ 뉴스 조회 api

▼ code

```
import requests
import os
from dotenv import load_dotenv

# 1. API 키 로딩
load_dotenv()
API_KEY = os.getenv("BIGKINDS_KEY")

# 2. 검색 파라미터 정의
query = "인공지능 OR AI"
from_date = "2025-04-01"
until_date = "2025-04-15"
provider = ["서울경제"]
preview_length = 200  # 기사 본문 미리보기 길이

# 3. 뉴스 검색 API 요청
search_url = "https://tools.kinds.or.kr/search/news"

search_payload = {
    "access_key": API_KEY,
    "argument": {
        "query": query,
        "published_at": {
            "from": from_date,
            "until": until_date
        },
        "provider": provider,
        "category": [],
        "category_incident": [],
        "provider_subject": [],
        "subject_info": [],
        "sort": { "date": "desc" },
        "return_from": 0,
        "return_size": 5,
        "fields": ["news_id", "title", "published_at"]
    }
}
```

```
search_response = requests.post(search_url, json=search_payload)

if search_response.status_code == 200:
    search_result = search_response.json()
    docs = search_result["return_object"]["documents"]
    total_hits = search_result["return_object"]["total_hits"]

    if not docs:
        print("❗ 뉴스 검색 결과가 없습니다.")
    else:
        print("✅ 뉴스 검색 API 호출 성공!")
        print(f"📦 총 검색 결과: {total_hits}건\n")

        # 뉴스 ID 추출
        news_ids = [doc["news_id"] for doc in docs]
        for i, doc in enumerate(docs, 1):
            print(f"{i}. 뉴스 ID: {doc['news_id']} | 제목: {doc['title']}")

        print("\n📥 뉴스 ID로 상세 기사 조회 시작...\n")

        # 4. 뉴스 조회 API 요청
        view_payload = {
            "access_key": API_KEY,
            "argument": {
                "news_ids": news_ids,
                "fields": [
                    "title",
                    "content",
                    "byline",
                    "category",
                    "published_at",
                    "provider",
                    "images",
                    "images_caption"
                ]
            }
        }

        view_response = requests.post(search_url, json=view_payload)
```

```
        if view_response.status_code == 200:
            view_result = view_response.json()
            print("✅ 뉴스 조회 API 호출 성공!\n")

            for i, doc in enumerate(view_result["return_object"]["documents"],
                print(f"----- {i}번째 뉴스 -----")
                print("제목:", doc.get("title", "없음"))
                print("기자:", doc.get("byline", "없음"))
                print("발행일:", doc.get("published_at", "없음"))
                print("카테고리:", doc.get("category", []))
                print("언론사:", doc.get("provider", "없음"))
                print("본문 (앞부분):", doc.get("content", "")[:preview_length], "...")

                # 이미지 있을 경우만 출력
                image_url = doc.get("images")
                if image_url:
                    print("🖼 이미지 URL:", image_url)
                    print("📝 이미지 설명:", doc.get("images_caption", "설명 없음"))
                print()

        else:
            print("❌ 뉴스 조회 API 오류:", view_response.status_code)
            try:
                print(view_response.json().get("reason", "알 수 없는 오류"))
            except:
                print(view_response.text)

else:
    print("❌ 뉴스 검색 API 오류:", search_response.status_code)
    try:
        print(search_response.json().get("reason", "알 수 없는 오류"))
    except:
        print(search_response.text)
```

▼ 항목 정리

📋 뉴스 조회 API (`/search/news`) 요청 항목 정리표

요청 필드	필수 여부	설명	예시
`access_key`	✅	발급받은 인증 키	`"abcde-12345..."`

news_ids	✅	조회할 뉴스 ID 리스트	["02100311.20250414103036001"]
fields	✅	받고 싶은 항목 리스트	["title", "content", "images"]

🧾 fields 에 넣을 수 있는 상세 항목

필드명	설명	예시
title	기사 제목	"AI 면접 시대"
content	기사 본문	"서울시는 AI 채용..."
published_at	발행일 (ISO 형식)	"2025-04-14T00:00:00.000+09:00"
byline	기자 이름	"홍길동"
provider	언론사명	"서울경제"
category	뉴스 카테고리 분류	["IT_과학>AI"]
images	이미지 파일 경로 (상대 경로)	"/02100311/2025/04/14/..."
images_caption	이미지 캡션 (설명 문구)	"AI 채용 체험 중"
provider_news_id	언론사 내부 기사 식별자	"20250414203036001"
publisher_code	뉴스진흥재단 기사 코드	"02100311"
provider_subject	주제 분류 (NewsML 기반)	[{ "subject_info": "사회" }]
subject_info ~ subject_info4	뉴스ML 주제 계층	"사회", "경제", etc.

📎 예시 요청 바디 (JSON)

```
{
  "access_key": "your_api_key",
  "argument": {
    "news_ids": [
      "02100311.20250414103036001",
      "02100311.20250413123017002"
    ],
    "fields": [
      "title",
      "content",
      "published_at",
      "images",
```

```
      "images_caption",
      "provider"
    ]
  }
}
```

🔍 응답 안에 어떤 구조로 오는지?

```
{
  "result": 0,
  "return_object": {
    "documents": [
      {
        "title": "...",
        "content": "...",
        "images": "...",
        "provider": "...",
        "published_at": "..."
      }
    ]
  }
}
```

▼ 오늘의 이슈 api(이슈 랭킹 api)

▼ code

```
import requests
import os
from dotenv import load_dotenv

# 1. 환경 변수에서 API 키 로딩
load_dotenv()
API_KEY = os.getenv("BIGKINDS_KEY")

# 2. 날짜 & 언론사 설정
url = "https://tools.kinds.or.kr/issue_ranking"
target_date = "2025-04-14"
provider = ["서울경제"]  # 생략 가능
```

```
# 3. 요청 바디 구성
payload = {
    "access_key": API_KEY,
    "argument": {
        "date": target_date,
        "provider": provider
    }
}

# 4. API 호출
response = requests.post(url, json=payload)

# 5. 결과 처리
if response.status_code == 200:
    result = response.json()
    topics = result["return_object"].get("topics", [])

    print(f"\n✅ 이슈 랭킹 API 호출 성공! ({target_date})\n")

    if not topics:
        print("📭 이 날은 이슈가 없어요. 조용한 날인가봐요.")
    else:
        # topic_rank 기준으로 오름차순 정렬 (중요도 높은 순)
        topics = sorted(topics, key=lambda x: x["topic_rank"])

        for i, topic in enumerate(topics, 1):
            print(f"🧠 {i}. 이슈 제목: {topic['topic']}")
            print(f"   🔗 주요 키워드: {topic['topic_keyword']}")
            print(f"   🔢 중요도 (topic_rank): {topic['topic_rank']}")
            print(f"   📰 관련 기사 수: {len(topic['news_cluster'])}")

            # 뉴스 ID 리스트 (선택 출력)
            print("   🆔 뉴스 ID:")
            for news_id in topic["news_cluster"]:
                print(f"     - {news_id}")
            print("-" * 80)

else:
    print("❌ 이슈 랭킹 API 오류 발생:", response.status_code)
```

```
    try:
        print("사유:", response.json().get("reason", "알 수 없는 오류"))
    except:
        print(response.text)
```

▼ 항목 정리

✅ 이슈 랭킹 API 요청 항목 정리표 (`/issue_ranking`)

요청 항목	필수 여부	설명	예시
`access_key`	✅	발급받은 API 인증 키	`"a1b2c3d4..."`
`argument.date`	✅	분석 대상 날짜 (YYYY-MM-DD)	`"2025-04-14"`
`argument.provider`	❌	특정 언론사 제한 (없으면 전체 대상)	`["서울경제"]`

📎 예시 요청 바디

```
{
  "access_key": "너의_API_KEY",
  "argument": {
    "date": "2025-04-14",
    "provider": ["서울경제"]
  }
}
```

🧠 참고 설명

항목	설명
`date`	1990년 1월 1일부터 오늘까지 지정 가능
`provider`	생략하면 전체 언론사 기준, 있으면 해당 언론사만 분석됨
`access_key`	인증이 안 되면 `Blank Access Key!` 에러 뜸

🔁 응답에서 받을 수 있는 데이터 구조

필드명	설명
`topic`	이슈 제목 (자동 생성됨)

`topic_keyword`	이슈에 포함된 주요 키워드들 (쉼표 구분)
`topic_rank`	중요도 (낮을수록 상위 이슈)
`news_cluster`	이슈에 포함된 뉴스 ID 리스트 (이걸로 뉴스 조회 가능)

예시 응답 구조

```
{
  "return_object": {
    "topics": [
      {
        "topic": "AI 채용 도입 가속화",
        "topic_rank": 1,
        "topic_keyword": "AI,면접,청년,역량,비대면",
        "news_cluster": [
          "02100311.20250414103036001",
          "02100311.20250414100522002"
        ]
      }
    ]
  }
}
```

💡 이 API의 핵심 활용법

- 이슈 → **요약 API 붙여서 자동 이슈 요약 가능**
- news_cluster → **조회 API로 본문 가져오기**
- topic_keyword → **워드클라우드나 연결 키워드 분석 가능**
- topic → **제목처럼 사용해서 카드뉴스, 대시보드 출력 가능**

▼ 연관어 분석 api(워드 클라우드)

▼ code

```
import requests
import os
from dotenv import load_dotenv

# 1. API 키 로딩
```

```
load_dotenv()
API_KEY = os.getenv("BIGKINDS_KEY")

# 2. 요청 설정
url = "https://tools.kinds.or.kr/word_cloud"

# 분석 기준 키워드와 기간
query_keyword = "인공지능"
from_date = "2025-04-01"
until_date = "2025-04-14"
provider = ["서울경제"]
category = ["IT_과학"]

# 3. 요청 바디 구성
payload = {
    "access_key": API_KEY,
    "argument": {
        "query": query_keyword,
        "published_at": {
            "from": from_date,
            "until": until_date
        },
        "provider": provider,
        "category": category,
        "category_incident": [],
        "byline": "",
        "provider_subject": []
    }
}

# 4. API 호출
response = requests.post(url, json=payload)

# 5. 결과 처리
if response.status_code == 200:
    result = response.json()
    nodes = result["return_object"].get("nodes", [])

    print(f"✅ 연관어 분석 API 호출 성공! (기준 키워드: '{query_keyword}')\n")
```

```
    if not nodes:
        print("📪 연관 키워드가 없습니다.")
    else:
        print("📌 연관 키워드 목록 (중요도 순):\n")
        # 중요도 기준 내림차순 정렬
        nodes = sorted(nodes, key=lambda x: x["weight"], reverse=True)

        for i, node in enumerate(nodes, 1):
            print(f"{i}. 🔑 키워드: {node['name']:<15} 중요도(weight): {node['wei
else:
    print("❌ 연관어 분석 API 오류 발생:", response.status_code)
    try:
        print("사유:", response.json().get("reason", "알 수 없는 오류"))
    except:
        print(response.text)
```

▼ 항목 정리

✅ 요청 URL

```
https://tools.kinds.or.kr/word_cloud
```

✅ 요청 항목 정리표 (`/word_cloud`)

요청 항목	필수 여부	설명	예시
`access_key`	✅	인증 키	`"abcde-12345..."`
`argument.query`	✅	연관어 분석 기준 키워드	`"인공지능"`
`argument.published_at.from`	✅	시작 날짜 (포함)	`"2025-04-01"`
`argument.published_at.until`	✅	종료 날짜 (포함)	`"2025-04-14"`
`argument.provider`	❌	언론사 제한	`["서울경제"]`
`argument.category`	❌	뉴스 분류 필터	`["IT_과학"]`
`argument.category_incident`	❌	사건/사고 분류 필터	`["범죄", "자연재해"]`
`argument.byline`	❌	기자 이름	`"홍길동"`
`argument.provider_subject`	❌	언론사 주제 분류 필터	`["경제", "정치"]`

🧠 구조 예시

```
{
  "access_key": "your_key",
  "argument": {
    "query": "인공지능",
    "published_at": {
      "from": "2025-04-01",
      "until": "2025-04-14"
    },
    "provider": ["서울경제"],
    "category": ["IT_과학"]
  }
}
```

✅ 응답 항목 (출력 구조)

항목	설명	예시
`nodes`	연관 키워드 리스트 배열	`[{"name": "면접", "weight": 1.33}]`
`name`	연관어	`"채용"`
`weight`	중요도 / 빈도 / 연결도 등 가중치	`0.67`
`id`	키워드 고유 ID	`3`
`level`	관계망 깊이 (보통 3)	`3` *(보통 무시해도 됨)*

📌 정리 요약

기능	설명
분석 기준 키워드	`"query": "인공지능"`
연관 키워드 갯수	보통 10~50개
활용 가능성	워드 클라우드, 키워드 그래프, 요약 키워드 추천

▼ 키워드 트렌드 api(뉴스 타임라인)

▼ code

```
import requests
import os
from dotenv import load_dotenv
```

```
import pandas as pd
import matplotlib.pyplot as plt

# 1. API 키 로딩
load_dotenv()
API_KEY = os.getenv("BIGKINDS_KEY")

# 2. 요청 설정
url = "https://tools.kinds.or.kr/time_line"
query_keyword = "인공지능"
from_date = "2025-03-01"
until_date = "2025-04-14"
provider = ["서울경제"]
interval = "day"  # day, month, year

# 3. 요청 바디 구성
payload = {
    "access_key": API_KEY,
    "argument": {
        "query": query_keyword,
        "published_at": {
            "from": from_date,
            "until": until_date
        },
        "provider": provider,
        "category": [],
        "category_incident": [],
        "byline": "",
        "provider_subject": [],
        "interval": interval,
        "normalize": "false"
    }
}

# 4. API 호출
response = requests.post(url, json=payload)

# 5. 결과 처리
if response.status_code == 200:
    result = response.json()
```

```
    timeline = result["return_object"].get("time_line", [])

    print(f"\n✅ 키워드 트렌드 API 호출 성공! (키워드: '{query_keyword}')\n")

    if not timeline:
        print("📭 기간 내 데이터 없음.")
    else:
        print("📊 날짜별 키워드 언급량:\n")
        dates = []
        counts = []
        for point in timeline:
            label = point["label"]
            count = point["hits"]
            print(f"{label} → {count}건")
            dates.append(label)
            counts.append(count)

        # ⬇ 시각화 (옵션)
        df = pd.DataFrame({"날짜": dates, "기사 수": counts})
        df["날짜"] = pd.to_datetime(df["날짜"], format="%Y%m%d")
        df.set_index("날짜", inplace=True)

        df.plot(kind="line", figsize=(12, 5), marker="o", title=f"'{query_keywor
        plt.ylabel("기사 수")
        plt.grid(True)
        plt.tight_layout()
        plt.show()

else:
    print("❌ 키워드 트렌드 API 오류:", response.status_code)
    try:
        print(response.json().get("reason", "알 수 없는 오류"))
    except:
        print(response.text)
```

▼ 항목 정리

📚 키워드 트렌드 API 기본 정보

항목	설명

API 목적	키워드가 뉴스에서 얼마나 자주 등장했는지 기간별로 집계
API URL	`https://tools.kinds.or.kr/time_line`
집계 단위	`"day"`, `"month"`, `"year"`
활용처	트렌드 그래프, 기간별 뉴스 분포, 인물 이슈 추이

✅ 요청 항목 정리표

항목	필수	설명	예시
`access_key`	✅	인증키	`"abcde-12345..."`
`query`	✅	분석할 키워드	`"인공지능"`
`published_at.from`	✅	시작일	`"2025-03-01"`
`published_at.until`	✅	종료일	`"2025-04-14"`
`interval`	✅	집계 단위 (`day`, `month`, `year`)	`"day"`
`normalize`	❌	정규화 여부 (boolean)	`"false"`
`provider`	❌	언론사 제한	`["서울경제"]`
`category`	❌	뉴스 분류	`["IT_과학"]`
`category_incident`	❌	사건/사고 분류	`["범죄"]`

📌 응답 형식 예시

```
"return_object": {
  "time_line": [
    { "label": "20250401", "hits": 12 },
    { "label": "20250402", "hits": 7 },
    ...
  ]
}
```

💡 활용 예시

사용 예	설명
인물/이슈 트렌드 분석	키워드: `"이재명"`, `"한미정상회담"` 등
트렌드 급등 시각화	갑자기 뜬 키워드 감지 (뉴스량 급등)
워드클라우드와 조합	많이 등장한 날 기준으로 키워드 분석

🎓 요약 정리

기능	뉴스 내 키워드 등장 횟수를 날짜별로 분석
핵심 요청	`query`, `from`, `until`, `interval`
출력	날짜(`label`), 건수(`hits`)
시각화	pandas + matplotlib 활용

▼ 인기검색어 api

▼ code

```
import requests
import os
from dotenv import load_dotenv

# 1. API 키 로딩
load_dotenv()
API_KEY = os.getenv("BIGKINDS_KEY")

# 2. 요청 설정
url = "https://tools.kinds.or.kr/query_rank"

from_date = "2025-04-01"
until_date = "2025-04-14"
offset = 10  # 가져올 인기 키워드 수

payload = {
    "access_key": API_KEY,
    "argument": {
        "from": from_date,
        "until": until_date,
        "offset": offset
        # "target_access_key": ""  ← 특정 기관 사용자 한정할 경우만 사용
    }
}

# 3. API 호출
response = requests.post(url, json=payload)

# 4. 결과 처리
if response.status_code == 200:
```

```
    result = response.json()
    keywords = result["return_object"].get("queries", [])

    print(f"✅ 인기검색어 API 호출 성공! ({from_date} ~ {until_date})\n")

    if not keywords:
        print("📪 인기 검색어가 없습니다.")
    else:
        print("📌 인기 검색어 목록:\n")
        for i, item in enumerate(keywords, 1):
            print(f"{i}. 🔍 키워드: {item['query']} | 🔢 검색 횟수: {item['count']}")
else:
    print("❌ 인기 검색어 API 오류 발생:", response.status_code)
    try:
        print("사유:", response.json().get("reason", "알 수 없는 오류"))
    except:
        print(response.text)
```

▼ 항목 정리

📚 인기검색어 API 개요

항목	설명
API 목적	**전체 사용자 or 특정 기관의 인기 검색 키워드 추출**
활용처	뉴스 큐레이션, 검색 추천어, 트렌드 분석
API URL	`https://tools.kinds.or.kr/query_rank`

✅ 요청 항목 정리

항목	필수	설명	예시
`access_key`	✅	인증키	`"abcde-12345..."`
`argument.from`	✅	조회 시작일	`"2025-04-01"`
`argument.until`	✅	조회 종료일	`"2025-04-15"`
`argument.offset`	✅	결과 개수 제한 (최대 인기검색어 수)	`10`
`argument.target_access_key`	❌	특정 사용자 기준 인기검색어만 보고 싶을 때 사용	(생략 가능)

🔁 예시 요청 바디

```
{
  "access_key": "your_api_key",
  "argument": {
    "from": "2025-04-01",
    "until": "2025-04-14",
    "offset": 10
  }
}
```

🧠 응답 예시 구조

```
{
  "return_object": {
    "queries": [
      {
        "date": "20250414",
        "query": "인공지능",
        "count": 152
      },
      {
        "date": "20250414",
        "query": "AI 면접",
        "count": 97
      }
    ]
  }
}
```

📌 응용 팁

활용 예시	설명
인기 키워드 기반 뉴스 추천	`query` 값을 다시 뉴스 검색 API에 넣어 연관 기사 뽑기
인기 키워드 트렌드 그래프	날짜별 인기 키워드의 변화 시각화 가능
키워드 기반 요약 or 워드클라우드	LLM 요약 프롬프트에 활용 가능

✅ 요약 정리표

목적	인기 검색 키워드 추출 (실사용자 기준)
요청 필드	`access_key` , `from` , `until` , `offset`
응답 필드	`query` (키워드), `count` (검색횟수), `date`
활용도	뉴스 트렌드, 요약 추천어, 워드 클라우드, LLM 요약 조건 등

▼ 뉴스 인용문 검색 api

▼ code

```
import requests
import os
from dotenv import load_dotenv

# 1. API 키 로딩
load_dotenv()
API_KEY = os.getenv("BIGKINDS_KEY")

# 2. 요청 URL
url = "https://tools.kinds.or.kr/search/quotation"

# 3. 요청 조건 설정
query = "인공지능"
from_date = "2025-04-01"
until_date = "2025-04-14"
provider = ["서울경제"]

# 4. 요청 바디 구성
payload = {
    "access_key": API_KEY,
    "argument": {
        "query": query,
        "published_at": {
            "from": from_date,
            "until": until_date
        },
        "provider": provider,
        "category": [],
        "category_incident": [],
        "byline": "",
```

```
        "provider_subject": [],
        "subject_info": [],
        "sort": { "date": "desc" },
        "hilight": 200,
        "return_from": 0,
        "return_size": 10,
        "fields": [
            "quotation",
            "provider",
            "byline",
            "published_at",
            "title"
        ]
    }
}

# 5. API 호출
response = requests.post(url, json=payload)

# 6. 결과 처리
if response.status_code == 200:
    result = response.json()
    docs = result["return_object"]["documents"]
    print(f"\n✅ 뉴스 인용문 API 호출 성공! (총 {len(docs)}건)\n")

    if not docs:
        print("📪 인용문 데이터 없음")
    else:
        for i, doc in enumerate(docs, 1):
            print(f"----- {i} -----")
            print("📰 제목:", doc.get("title", "없음"))
            print("🗒️ 발언:", doc.get("quotation", "❌ 없음"))
            print("👤 기자:", doc.get("byline", "없음"))
            print("🗓️ 발행일:", doc.get("published_at", "없음"))
            print("📰 언론사:", doc.get("provider", "없음"))
            print()
else:
    print("❌ 뉴스 인용문 API 오류 발생:", response.status_code)
    try:
        print(response.json().get("reason", "알 수 없는 오류"))
```

```
    except:
        print(response.text)
```

▼ 항목 정리

📌 요청 URL

```
https://tools.kinds.or.kr/search/quotation
```

✅ 요청 항목 정리표

항목	필수	설명	예시
`access_key`	✅	인증키	`"abcde-12345..."`
`argument.query`	✅	인용문 기준 키워드	`"AI"`
`argument.published_at.from`	✅	시작일	`"2025-04-01"`
`argument.published_at.until`	✅	종료일	`"2025-04-14"`
`argument.provider`	❌	언론사 제한	`["서울경제"]`
`argument.category`	❌	분류 제한	`["정치"]`
`argument.category_incident`	❌	사건/사고 필터	`["범죄"]`
`argument.byline`	❌	기자 이름	`"홍길동"`
`argument.provider_subject`	❌	주제 분류	`["경제"]`
`return_size`	❌	반환 개수 제한 (최대 100)	`10`
`fields`	✅	받고 싶은 필드 지정	`["quotation", "provider", "byline"]` 등

📦 응답 예시

```
{
  "title": "AI 윤리, 국회 토론회 열려",
  "quotation": "윤 의원은 'AI는 인간보다 낫지 않다'고 말했다.",
  "byline": "홍길동",
  "published_at": "2025-04-12",
  "provider": "서울경제"
}
```

💡 활용 예시

기능	설명
발언 중심 뉴스 요약	정치·사법 기사 분석용
발언자 중심 리포트	누가 어떤 사안에서 자주 언급되는지 추출
인물-이슈 매핑	`"이재명"` 관련 발언 자동 수집 등

🎓 요약 정리

항목	의미
목적	**뉴스에서 발언/인용만 뽑아내기**
핵심 필드	`"quotation"`, `"byline"`, `"provider"`
활용처	정치/사회/경제 인용 요약 리포트, 기자별 분석, 발언 감성 분석 등

▼ **오늘의 키워드 api(분야별 키워드)**

▼ code

```
import requests
import os
from dotenv import load_dotenv

# 1. API 키 로딩
load_dotenv()
API_KEY = os.getenv("BIGKINDS_KEY")
url = "https://tools.kinds.or.kr/today_category_keyword"

# 2. 요청 바디 구성
payload = {
    "access_key": API_KEY,
    "argument": {}
}

# 3. API 호출
response = requests.post(url, json=payload)

# 4. 결과 처리
if response.status_code == 200:
    result = response.json()
```

```
    print("✅ 오늘의 키워드 API 호출 성공!\n")

    print("📊 분야별 기사 비중 (cate_ratio):\n")
    for item in result["return_object"].get("cate_ratio", []):
        print(f"🗂 분야: {item['category_name']}")
        print(f"   📈 비중: {item['category_percent']}%")
        print(f"   📰 기사 수: {item['category_count']}건\n")

    print("🔑 분야별 주요 키워드 (cate_keyword):\n")
    for item in result["return_object"].get("cate_keyword", []):
        print(f"📌 분야: {item['category_name']}")
        print(f"   🔍 키워드: {item['named_entity']}")
        print(f"   📊 등장 횟수: {item['named_entity_count']}")
        print(f"   🧠 타입: {item['named_entity_type']} | 단계: {item['entity_step'
        print("-" * 50)

else:
    print("❌ 오늘의 키워드 API 오류:", response.status_code)
    try:
        print("사유:", response.json().get("reason", "알 수 없는 오류"))
    except:
        print(response.text)
```

▼ 항목 정리

🧠 기능 요약

> "정치", "경제", "사회" 같은 카테고리별로
>
> 오늘 뉴스에서 많이 등장한 **주요 키워드 (고유명사 등)**를
>
> **개체명 기준**으로 분석해주는 API

📌 요청 URL

```
https://tools.kinds.or.kr/today_category_keyword
```

✅ 요청 항목 정리

항목	필수	설명	예시
`access_key`	✅	인증키	`"abcde-12345..."`
`argument`	❌	따로 설정 안 해도 됨 (기본값 전체 범위)	`{}`

※ 오늘 날짜가 자동으로 설정됨

※ 날짜/카테고리 직접 지정 불가 (완전 자동화 API)

✅ 응답 항목 정리

▶ `cate_ratio` 리스트 (카테고리 비중 정보)

필드명	설명	예시
`category_name`	분야 이름	`"경제"`
`category_percent`	전체 기사 중 비율 (%)	`14.3`
`category_count`	해당 분야 기사 수	`317`

▶ `cate_keyword` 리스트 (핵심 키워드 정보)

필드명	설명	예시
`category_name`	분야	`"정치"`
`named_entity`	키워드	`"윤석열"`
`named_entity_count`	등장 횟수	`56`
`named_entity_type`	개체 유형	`"PS"` → 사람 (인물)
`entity_step`	중요도 단계 (step1~5)	`"step4"` (많이 등장할수록 높음)

> entity_type 종류:
>
> `PS` : 인물 / `OG` : 기관 / `LC` : 장소

💡 활용 팁

활용 아이디어	설명
대시보드용	분야별 관심도와 키워드를 한 화면에
뉴스 추천 알고리즘	`"정치"` 분야에서 `"윤석열"`이 뜨면 관련 기사 추천
카드뉴스 자동화	키워드만으로 주요 분야 요약 생성 가능

보고서 슬라이드	분야별 키워드 랭킹을 슬라이드로 구성

🎓 요약 정리

목적	오늘 하루 뉴스에서 분야별 핵심 키워드 및 기사 분포 확인
주요 필드	`cate_ratio`, `cate_keyword`
자동 날짜 적용	오늘 기준
분석 단위	`"정치"`, `"경제"`, `"사회"`, `"문화"` 등 분야별 분리
개체명 타입	인물(PS), 기관(OG), 장소(LC) 등

▼ 특성 추출 api

▼ code

```
import requests
import os
from dotenv import load_dotenv

# 1. 환경 변수 로딩
load_dotenv()
API_KEY = os.getenv("BIGKINDS_KEY")

# 2. 요청 URL
url = "https://tools.kinds.or.kr/feature"

# 3. 분석할 기사 텍스트 입력
payload = {
    "access_key": API_KEY,
    "argument": {
        "title": "AI 채용 시대... 서울시, 비대면 면접 체험 확대",
        "sub_title": "AI 면접 역량 검사 프로그램, 청년 구직자 대상 운영",
        "content": """
        서울시는 청년들이 급변하는 취업 트렌드에 발맞춰 비대면 AI 채용에 대비할 수 있
        AI 면접 체험과 역량 검사 프로그램을 제공한다.
        이 프로그램은 지난해 1만 5506명의 청년 구직자가 이용했으며 95.8%의 만족도
        """
    }
}

# 4. API 호출
```

```
response = requests.post(url, json=payload)

# 5. 결과 처리
if response.status_code == 200:
    result = response.json()
    print("✅ 특성 추출 API 호출 성공!\n")

    for field in ["title", "sub_title", "content"]:
        print(f"📌 {field.upper()} 키워드:")
        text = result["return_object"]["result"].get(field, "")

        if not text.strip():
            print(" - (데이터 없음)")
            continue

        for item in text.strip().split(" "):  # 공백으로 나눔
            if "|" in item:
                keyword, score = item.split("|")
                print(f" - 🔑 {keyword} (점수: {score})")
            else:
                print(f" - 🔑 {item} (점수 없음)")
        print()

else:
    print("❌ 특성 추출 API 오류:", response.status_code)
    try:
        print(response.json().get("reason", "알 수 없는 오류"))
    except:
        print(response.text)
```

▼ 항목 정리

✅ [최종 정리] OpenAPI 10번 – 특성 추출 API

📌 API 개요

항목	설명
API 이름	특성 추출 API
요청 URL	https://tools.kinds.or.kr/feature

기능 설명	뉴스 기사에서 **핵심 키워드와 중요도 점수**를 추출 (title / sub_title / content별 분리)
활용 목적	자동 태깅, 요약 전 키워드 설정, LLM 프롬프트 강화, 카드뉴스 제목 자동화 등

🧠 요청 항목

필드명	필수	설명
`access_key`	✅	인증 API 키
`title`	✅	기사 제목
`sub_title`	✅	부제목 (없으면 빈 문자열 가능)
`content`	✅	본문 텍스트 (한 문단 이상 추천)

📌 참고: 텍스트 안에 쌍따옴표 `"` 들어가면 백슬래시 `\"`로 escape 필요

🧾 응답 구조

응답은 **각 영역별로 하나의 긴 문자열**로 반환됨.

→ 각 키워드는 `"키워드|점수"` 형식이며, 이걸 `" "` (공백) 기준으로 나눠야 함.

```
{
  "result": 0,
  "return_object": {
    "result": {
      "title": "AI|0.12 채용|0.11 서울시|0.10",
      "sub_title": "면접|0.09 프로그램|0.08",
      "content": "청년|0.15 구직자|0.13 비대면|0.12"
    }
  }
}
```

🧠 파싱 주의 사항

문제	해결 방법
점수 없이 키워드만 있는 경우	`"
전체 응답은 문자열 하나	`.split(" ") → .split("
예외 대응 필수	`try: ... except:` 또는 `if "

🛠️ 활용 코드 요약

```
for field in ["title", "sub_title", "content"]:
    text = result["return_object"]["result"].get(field, "")
    for item in text.strip().split(" "):
        if "|" in item:
            keyword, score = item.split("|")
            print(f"{keyword} ({score})")
        else:
            print(f"{item} (점수 없음)")
```

💡 실전 활용 팁

기능	설명
자동 태그 생성	LLM 없이도 키워드로 태그 자동화 가능
요약 전 프롬프트 강화	`"다음 키워드를 중심으로 요약해줘: [AI, 채용, 청년]"`
슬라이드 제목 자동 생성	title/sub_title 키워드 조합으로 3~5자 요약 제목 만들기
유사 기사 클러스터링	content 키워드 벡터화해서 비슷한 기사 묶기 가능

- 기사 → 키워드 추출 → 태그 적용
- 기사 → 키워드 추출 → 요약 프롬프트 강화
- 기사 → 키워드 추출 → 연관어 or 트렌드 API와 연동

▼ 키워드 추출 api

▼ code

```
import requests
import os
from dotenv import load_dotenv

# 1. 환경 변수 로딩
load_dotenv()
API_KEY = os.getenv("BIGKINDS_KEY")

# 2. API URL
url = "https://tools.kinds.or.kr/keyword"

# 3. 본문 입력
payload = {
```

```
    "access_key": API_KEY,
    "argument": {
        "title": "AI 채용 시대... 서울시, 비대면 면접 체험 확대",
        "sub_title": "AI 면접 역량 검사 프로그램, 청년 구직자 대상 운영",
        "content": """
        서울시는 청년들이 급변하는 취업 트렌드에 발맞춰 비대면 AI 채용에 대비할 수 있
        AI 면접 체험과 역량 검사 프로그램을 제공한다.
        """
    }
}

# 4. API 호출
response = requests.post(url, json=payload)

# 5. 결과 출력
if response.status_code == 200:
    result = response.json()
    print("✅ 키워드 추출 API 호출 성공!\n")

    for field in ["title", "sub_title", "content"]:
        keywords = result["return_object"]["result"].get(field, "")
        print(f"📌 {field.upper()} 키워드:")
        for word in keywords.strip().split(" "):
            if word:
                print(f" - 🔑 {word}")
        print()
else:
    print("❌ 키워드 추출 API 오류:", response.status_code)
    try:
        print(response.json().get("reason", "알 수 없는 오류"))
    except:
        print(response.text)
```

▼ 항목 정리

🔍 API 개요

항목	설명
API 이름	키워드 추출 API
요청 URL	`https://tools.kinds.or.kr/keyword`

기능	기사 내 **title**, **sub_title**, **content**에서 키워드 추출
특이사항	결과는 문자열 (공백 구분), score 정보 없음

📄 요청 형식

```
{
  "access_key": "your_api_key",
  "argument": {
    "title": "AI 채용 시대",
    "sub_title": "비대면 면접 증가",
    "content": "서울시는 AI 면접 체험과 역량 검사를 제공한다."
  }
}
```

✅ 응답 예시

```
{
  "result": 0,
  "return_object": {
    "result": {
      "title": "AI 채용 시대",
      "sub_title": "비대면 면접 증가",
      "content": "서울시 AI 면접 체험 역량 검사 제공"
    }
  }
}
```

💡 활용 요약

항목	설명
`title`	제목에서 핵심 키워드 추출
`sub_title`	부제목 키워드
`content`	본문 키워드
형식	`"단어 단어 단어"` → 공백으로 나눠짐
점수 정보	❌ 없음 (순위만 있음)

활용 예시

사용처	설명
키워드 기반 검색	CMS에 자동 태그로 활용
카드뉴스 자동 제목	title/sub_title 키워드 조합
AI 요약 강화	추출 키워드를 LLM 프롬프트에 삽입

▼ topn 키워드 api

▼ code

```
import requests
import os
from dotenv import load_dotenv

# API 키 로딩
load_dotenv()
API_KEY = os.getenv("BIGKINDS_KEY")

url = "https://tools.kinds.or.kr/topn_keyword"

payload = {
    "access_key": API_KEY,
    "argument": {
        "published_at": {
            "from": "2025-04-01",
            "until": "2025-04-14"
        },
        "query": "인공지능 OR AI"
    }
}

response = requests.post(url, json=payload)

if response.status_code == 200:
    result = response.json()

    if result.get("result") == 0:
        keywords = result["return_object"]["result"]
        print("✅ TopN 키워드 API 호출 성공!\n")
        if not keywords:
```

```
            print("📪 추출된 키워드가 없습니다.")
        else:
            for item in keywords:
                print(f"{item['rank']}. 🔑 {item['name']}")
    else:
        print("⚠️ API 내부 실패")
        print("사유:", result.get("reason", "알 수 없는 오류"))
else:
    print("❌ API 통신 오류:", response.status_code)
    try:
        print(response.json().get("reason", "알 수 없는 오류"))
    except:
        print(response.text)
```

▼ 항목 정리

✅ [최종 정리] TopN 키워드 API

🧠 목적

> 설정한 기간 or 시점을 기준으로
>
> 뉴스 기사에서 가장 많이 등장한 키워드를 **순위로 뽑아줌**

📌 API 정보

항목	내용
API 이름	TopN 키워드 API
API URL	`https://tools.kinds.or.kr/topn_keyword`
요청 방식	POST (JSON)
출력 형식	키워드 랭킹 리스트 (`rank` , `name`)
점수 제공	❌ 없음 (단순 랭킹만 반환)

🧾 요청 방법 2가지

✅ 유형 1: 특정 시점 기준 24시간

```
{
  "access_key": "YOUR_KEY",
  "argument": {
    "date_hour": "2025041510"  // yyyyMMddHH
  }
}
```

→ `"2025-04-15 10:00"`부터 24시간 전까지 뉴스 대상

✅ 유형 2: 특정 기간 + 검색어 조합

```
{
  "access_key": "YOUR_KEY",
  "argument": {
    "published_at": {
      "from": "2025-04-01",
      "until": "2025-04-14"
    },
    "query": "인공지능 OR AI"
  }
}
```

> 💥 주의:
>
> `category` 필드는 문서에는 나와 있지만 **현재 API에서는 지원되지 않음**
>
> 입력 시 오류 발생: `허용되지 않은 category 입력값입니다.`

✅ 응답 예시

```
{
  "result": 0,
  "return_object": {
    "result": [
      { "rank": "1", "name": "AI" },
      { "rank": "2", "name": "윤석열" },
      { "rank": "3", "name": "반도체" }
```

```
    ]
  }
}
```

필드명	설명
`rank`	순위 (문자열)
`name`	키워드

※ 점수(score)는 없음. 단순 등장 횟수 기반 순위 추정.

🛠️ 실전 코드 (유형2 기준)

```
import requests, os
from dotenv import load_dotenv

load_dotenv()
API_KEY = os.getenv("BIGKINDS_KEY")
url = "https://tools.kinds.or.kr/topn_keyword"

payload = {
    "access_key": API_KEY,
    "argument": {
        "published_at": {
            "from": "2025-04-01",
            "until": "2025-04-14"
        },
        "query": "인공지능 OR AI"
    }
}

response = requests.post(url, json=payload)

if response.status_code == 200:
    result = response.json()
    if result.get("result") == 0:
        keywords = result["return_object"]["result"]
        print("✅ TopN 키워드 API 호출 성공!\n")
        for item in keywords:
            print(f"{item['rank']}. 🔑 {item['name']}")
    else:
```

```
        print("⚠️ 내부 API 에러:", result.get("reason", "사유 없음"))
else:
    print("❌ 통신 오류:", response.status_code)
```

💡 실전 활용 아이디어

기능	설명
키워드 기반 요약 프롬프트	`"아래 주요 키워드를 중심으로 요약하라"`
자동 태그	기사 요약 후 태그 자동 입력
대시보드용 그래프	랭킹 기반으로 막대/워드클라우드 시각화
트렌드 비교	이전 주/월 대비 TopN 변화 비교

🎯 핵심 체크리스트

항목	체크
✅ `query`	`"인공지능"` 등 명확히 지정
⛔ `category`	사용하지 말 것 (오류 발생)
✅ `published_at` or `date_hour`	날짜 형식 맞춰서 정확하게
❌ `score`, `count` 없음	**오로지 `rank`, `name` 만 제공**

✅ 최종 요약

항목	설명
이름	TopN 키워드 API
URL	`https://tools.kinds.or.kr/topn_keyword`
기능	기간 내 가장 자주 언급된 키워드 목록
결과	`rank`, `name` 만 제공
활용	요약/태깅/분석/시각화 전처리용으로 훌륭함