---

- 출처: LangChain 공식 문서 또는 해당 교재명
- 원본 URL: https://smith.langchain.com/hub/teddynote/summary-stuff-documents

---

## 재귀적 JSON 분할(RecursiveJsonSplitter)

### RecursiveJsonSplitter

- **해당 JSON 분할기**
  - **JSON 데이터를 깊이 우선 탐색(depth-first traversal) → 더 작은 JSON 청크(chunk)를 생성**
  - **중첩된 JSON 객체를 가능한 한 유지하려고 시도**
    - **필요한 경우 객체 분할**: 청크의 크기를 **`min_chunk_size`와 `max_chunk_size` 사이로 유지하기 위해 필요한 경우
    - **분할되지 않는 경우**: 값이 중첩된 JSON이 아니라 매우 큰 문자열인 경우
  - 청크 크기에 대한 엄격한 제한이 필요한 경우: 이 분할기 사용 이후 → `Recursive Text Splitter`를 사용 → 해당 청크를 처리하는 것을 고려 가능

- **분할 기준**
  - 텍스트 분할 방식: **JSON 값 기준**
  - 청크 크기 측정 방식: **문자 수 기준**

---

- 사전에 `VS Code` 터미널에 설치할 것

```
pip install -qU langchain-text-splitters
```

---

- `requests.get()` 함수 사용 → *"https://api.smith.langchain.com/openapi.json"* URL에서 JSON 데이터 가져오기

- 가져온 `JSON` 데이터 → `json()` 메서드 → **`Python`** 딕셔너리 형태로 변환 → **`json_data`** 변수에 저장

```
import requests

# JSON 데이터 가져오기
json_data = requests.get("https://api.smith.langchain.com/openapi.json").json()
```

```
print(type(json_data))          # <class 'dict'>
```

- **`RecursiveJsonSplitter`** 사용 → **`JSON`** 데이터 분할하기

```
from langchain_text_splitters import RecursiveJsonSplitter

# JSON 데이터를 최대 300 크기의 청크로 분할하는 RecursiveJsonSplitter 객체 생성하기
splitter = RecursiveJsonSplitter(max_chunk_size=300)
```

- **`splitter.split_json()`** 함수 사용 → **`JSON`** 데이터를 **`재귀적으로`** 분할

```
# JSON 데이터 재귀적으로 분할하기
# 작은 JSON 조각에 접근하거나 조작해야 하는 경우에 사용하기
json_chunks = splitter.split_json(json_data=json_data)
```

- **`splitter.create_documents()`** 메서드 사용 → **`JSON`** 데이터를 문서 형식으로 변환
- **`splitter.split_text()`** 메서드 사용 → **`JSON`** 데이터 문자열 **`리스트`**로 분할

```
# JSON 데이터를 기반으로 문서 생성하기
docs = splitter.create_documents(texts=[json_data])

# JSON 데이터를 기반으로 문자열 청크 생성하기
texts = splitter.split_text(json_data=json_data)

# 첫 번째 문자열 출력해보기
print(docs[0].page_content)

print("\n", "===" * 20, "\n")

# 분할된 문자열 청크 출력해보기
print(texts[0])
```

- 셀 출력

```
{"openapi": "3.1.0", "info": {"title": "LangSmith", "description": "The LangSmi

============================================================
```

```
{"openapi": "3.1.0", "info": {"title": "LangSmith", "description": "The LangSmi
```

```
# 청크의 크기 확인해보기

print([len(text) for text in texts][:10])
```

- 셀 출력

```
[404, 244, 238, 344, 207, 227, 224, 231, 126, 646]
```

- **`texts[9]`을 출력하여 큰 청크 중 하나를 검토한 결과, 해당 청크에 리스트 객체가 포함되어 있음을 확인**
  - 9번째 청크의 크기가 제한(300)을 초과한 이유: **리스트 객체이기 때문**
  - 이는 `RecursiveJsonSplitter`가 리스트 객체는 분할하지 않기 때문

```
print(texts[2])
```

```
print(texts[9])
```

- 출력: `texts[2]`

```
{"paths": {"/api/v1/sessions/{session_id}/dashboard": {"post": {"operationId": "
```

---

- 출력: `texts[9]`

```
{"paths": {"/api/v1/sessions/{session_id}": {"get": {"parameters": [{"name": "s
```

- **`2, 9번 index 청크` 파싱하기 ← `json` 모듈 사용**

```
import json

json_data2 = json.loads(texts[2])
json_data2["paths"]
```

```
import json

json_data = json.loads(texts[9])
json_data["paths"]
```

- 출력: `texts[2]`

```
{'/api/v1/sessions/{session_id}': {'get': {'parameters': {'2': {'name': 'stats_
 'in': 'query',
 'required': False,
 'schema': {'anyOf': {'0': {'type': 'string', 'format': 'date-time'},
   '1': {'type': 'null'}},
  'title': 'Stats Start Time'}}}}}}
```

- 출력: `texts[9]`

```
{'/api/v1/sessions/{session_id}': {'get': {'parameters': [{'name': 'session_id'
    'in': 'path',
    'required': True,
    'schema': {'type': 'string', 'format': 'uuid', 'title': 'Session Id'}},
    {'name': 'include_stats',
    'in': 'query',
    'required': False,
    'schema': {'type': 'boolean',
    'default': False,
    'title': 'Include Stats'}},
    {'name': 'stats_start_time',
    'in': 'query',
    'required': False,
    'schema': {'anyOf': [{'type': 'string', 'format': 'date-time'},
    {'type': 'null'}],
    'title': 'Stats Start Time'}},
    {'name': 'accept',
    'in': 'header',
    'required': False,
    'schema': {'anyOf': [{'type': 'string'}, {'type': 'null'}],
    'title': 'Accept'}}]}}}
```

- **`convert_lists` 매개변수를 `True`로 설정 → `JSON` 내의 `리스트`를 `index:item 형태`의 `key:value` 쌍으로 변환**

```
# 다음은 JSON을 전처리하고 리스트를 인덱스:항목을 키:값 쌍으로 하는 딕셔너리로 변환하기

texts = splitter.split_text(json_data=json_data, convert_lists=True)          # 딕
```

```
# 청크의 크기 확인해보기
print([len(text) for text in texts][:10])
```

- 청크 크기 감소, 숫자도 감소

```
[203, 212, 263, 221]
```

```
# 리스트 → 딕셔너리로 변환
# 결과 확인해보기
print(texts[2])
```

- 셀 출력 (`dict`로 변환된 `texts[2]`)

```
{"paths": {"/api/v1/sessions/{session_id}": {"get": {"parameters": {"2": {"name"
```

- **`docs` 리스트의 특정 인덱스에 해당하는 문서 확인해보기**

```
# 2번 문서 확인해보기

docs[2]
```

- 셀 출력: `texts[2]`의 문서 출력

```
Document(metadata={}, page_content='{"paths": {"/api/v1/sessions/{session_id}/da
```

---

- *next:* ***`CH08 임베딩 (Embedding)`***

---