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

---

마크다운 헤더 텍스트 분할 (MarkdownHeaderTextSplitter)

- 마크다운 파일의 구조를 이해하고 효율적으로 다루는 것 = 문서 작업에 있어 매우 중요
  - 특히, 문서의 전체적인 맥락과 구조를 고려하여 의미 있는 방식으로 텍스트를 임베딩하는 과정 = 광범위한 의미와 주제를 더 잘 포착할 수 있는 포괄적인 벡터 표현을 생성하는 데 큰 도움
  - **마크다운 파일의 특정 부분으로 내용을 나누는 것 = 텍스트의 공통된 맥락을 유지하면서도, 문서의 구조적 요소를 효과적으로 활용하려는 시도**
    - 예시: 헤더별로 내용을 나누고 싶은 경우
    - 문서 내에서 각각의 헤더 아래에 있는 내용을 기반으로 서로 연관된 정보 덩어리, 즉 청크를 만들고 싶은 경우

MarkdownHeaderTextSplitter

- **MarkdownHeaderTextSplitter** 활용
  - 문서를 지정된 헤더 집합에 따라 분할 → **각 헤더 그룹 아래의 내용을 별도의 청크로 관리할 수 있게 함**
  - 문서의 전반적인 구조를 유지하면서도 내용을 더 세밀하게 다룰 수 있게 되며, 이는 다양한 처리 과정에서 유용하게 활용

- 사전에 VS Code 터미널에 설치할 것

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

- **MarkdownHeaderTextSplitter 사용 → 마크다운 형식의 텍스트를 헤더 단위로 분할**
  - 마크다운 문서의 **헤더(#, ##, ### 등)를 기준** = 텍스트를 분할하는 역할
  - markdown_document 변수 = 마크다운 형식의 문서 할당

- `headers_to_split_on` 리스트 = `마크다운 헤더 레벨`과 `해당 레벨의 이름`이 **`튜플 형태`로 정의***
- `MarkdownHeaderTextSplitter` 클래스 사용 → `markdown_splitter` 객체 생성 → **`headers_to_split_on` 매개변수로 `분할 기준`이 되는 `헤더 레벨` 전달**
- `split_text` 메서드 호출 → **`markdown_document`를 헤더 레벨에 따라 분할**

```
from langchain_text_splitters import MarkdownHeaderTextSplitter

# 마크다운 형식의 문서를 문자열로 정의하기
markdown_document = "# Title\n\n## 1. SubTitle\n\nHi this is Jim\n\nHi this is Jo
print(markdown_document)
```

- 셀 출력

```
# Title

## 1. SubTitle

Hi this is Jim

Hi this is Joe

### 1-1. Sub-SubTitle

Hi this is Lance

## 2. Baz

Hi this is Molly
```

```
# 문서를 분할할 헤더 레벨과 해당 레벨의 이름을 정의하기
headers_to_split_on = [
    (
        "#",                # 헤더 레벨 1은 '#'로 표시, 'Header 1'이라는 이름
        "Header 1",
    ),
    (
        "##",               # 헤더 레벨 2는 '##'로 표시, 'Header 2'라는 이름
        "Header 2",
    ),
    (
        "###",              # 헤더 레벨 3은 '###'로 표시, 'Header 3'이라는 이름
        "Header 3",
    ),
]

# 마크다운 헤더를 기준으로 텍스트를 분할하는 MarkdownHeaderTextSplitter 객체 생성하기
markdown_splitter = MarkdownHeaderTextSplitter(headers_to_split_on=headers_to_spl
```

```
# markdown_document를 헤더를 기준으로 분할하여 md_header_splits에 저장하기
md_header_splits = markdown_splitter.split_text(markdown_document)


# 분할된 결과를 출력하기
for header in md_header_splits:
    print(f"{header.page_content}")
    print(f"{header.metadata}", end="\n====================\n")
```

- 셀 출력

```
Hi this is Jim
Hi this is Joe
{'Header 1': 'Title', 'Header 2': '1. SubTitle'}
====================
Hi this is Lance
{'Header 1': 'Title', 'Header 2': '1. SubTitle', 'Header 3': '1-1. Sub-SubTitle
====================
Hi this is Molly
{'Header 1': 'Title', 'Header 2': '2. Baz'}
====================
```

각 마크다운 그룹 내 = `원하는 텍스트 분할기(text splitter)` 적용 가능

```
from langchain_text_splitters import RecursiveCharacterTextSplitter

markdown_document = "# Intro \n\n## History \n\nMarkdown[9] is a lightweight mark

print(markdown_document)
```

- 셀 출력

```
# Intro

## History

Markdown[9] is a lightweight markup language for creating formatted text using a

Markdown is widely used in blogging, instant messaging, online forums, collabora

## Rise and divergence

As Markdown popularity grew rapidly, many Markdown implementations appeared, dri
```

```
additional features such as tables, footnotes, definition lists,[note 1] and Ma

#### Standardization

From 2012, a group of people, including Jeff Atwood and John MacFarlane, launch

## Implementations

Implementations of Markdown are available for over a dozen programming language
```

- `MarkdownHeaderTextSplitter` 사용 → 마크다운 문서를 `헤더`를 `기준`으로 `분할`

```
# 분할할 헤더 레벨과 해당 레벨의 이름을 지정하기
headers_to_split_on = [
    ("#", "Header 1"),
    ("##", "Header 2"),
]

# Markdown 문서를 헤더 레벨에 따라 분할하기
markdown_splitter = MarkdownHeaderTextSplitter(
    headers_to_split_on=headers_to_split_on, strip_headers=False
)

# 마크다운 문서를 헤더를 기준으로 분할하기
md_header_splits = markdown_splitter.split_text(markdown_document)

# 분할된 결과 출력하기
for header in md_header_splits:
    print(f"{header.page_content}")
    print(f"{header.metadata}", end="\n====================\n")
```

- 셀 출력

```
# Intro
## History
Markdown[9] is a lightweight markup language for creating formatted text using a
Markdown is widely used in blogging, instant messaging, online forums, collabora
{'Header 1': 'Intro', 'Header 2': 'History'}
====================
## Rise and divergence
As Markdown popularity grew rapidly, many Markdown implementations appeared, dri
additional features such as tables, footnotes, definition lists,[note 1] and Ma
#### Standardization
From 2012, a group of people, including Jeff Atwood and John MacFarlane, launch
{'Header 1': 'Intro', 'Header 2': 'Rise and divergence'}
====================
```

```
## Implementations
Implementations of Markdown are available for over a dozen programming language
{'Header 1': 'Intro', 'Header 2': 'Implementations'}
====================
```

- *next:* ***`HTML 헤더 텍스트 분할 (HTMLHeaderTextSplitter)`***
