

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


▼

PDF

- **PDF**
 - ISO 32000으로 표준화된 파일 형식
 - Adobe가 1992년에 문서를 제시하기 위해 개발
 - 응용 소프트웨어, 하드웨어 및 운영 시스템에 독립적인 방식으로 텍스트 서식 및 이미지를 포함
- **LangChain Document 형식 로드 방법 가이드**
 - 다운스킴에서 사용됨
 - 다양한 PDF 파서와 통합
 - 일부: 간단하고 상대적으로 저수준
 - 다른 일부: OCR 및 이미지 처리를 지원하거나 고급 문서 레이아웃 분석 수행
 - 사용자의 애플리케이션에 따라 선택 달라짐
 - **LangChain Document**

▼

AutoRAG 팀에서의 PDF 실험

- **AutoRAG 에서 진행한 실험을 토대로 작성한 순위표**
- 아래 표기된 숫자 = 등수 (The lower, the better)
 -  AutoRAG 에서 진행한 실험을 토대로 작성한 순위표 등수
 - 출처: [AutoRAG Medium 블로그](#)

API KEY를 환경변수로 관리하기 위한 설정 파일

```
import os
```

```
from dotenv import load_dotenv
```

```
# API KEY 정보로드
load_dotenv()
```

```
# true
```



실습에 활용한 문서

- 출처: 소프트웨어정책연구소(SPRI) - 2023년 12월호
 - 저자: 유재홍(AI정책연구실 책임연구원), 이지수(AI정책연구실 위촉연구원)
 - 위치
 - 링크: <https://spri.kr/posts/view/23669>
 - 파일: `../06../data/`
 - 파일명: SPRI_AI_Brief_2023년12월호_F.pdf

```
# data 폴더에서 다운로드 받기
```

```
FILE_PATH = "../06_Document_Loader/data/SPRI_AI_Brief_2023년12월호_F.pdf"
```

```
# 문서 객체(docs)의 메타데이터 깔끔하게 출력하는 함수 정의하기
```

```
def show_metadata(docs):
    # 문서(docs) 리스트가 비어있지 않은지 확인하기
    if docs:
        print("[metadata]")
        # 첫 번째 문서의 모든 메타데이터 키 출력
        print(list(docs[0].metadata.keys()))
        print("\n[examples]")
        # 메타데이터 키 중 가장 긴 키의 길이를 찾아서 정렬에 사용
        max_key_length = max(len(k) for k in docs[0].metadata.keys())
        # 첫 번째 문서의 메타데이터 항목들을 순회하며 출력
        for k, v in docs[0].metadata.items():
            # 키는 왼쪽 정렬하고, 가장 긴 키의 길이에 맞춰 간격 띄우기
            print(f"{k:<{max_key_length}} : {v}")
```



PyPDF

- `pypdf` 사용 → PDF를 문서 배열로 로드
- 각 문서는 page 번호와 함께 페이지 내용 및 메타데이터를 포함
 - 먼저 터미널에 설치
 -

```
pip install -qU pypdf
```

```

from langchain_community.document_loaders import PyPDFLoader

# 파일 경로 설정
loader = PyPDFLoader(FILE_PATH)

# PDF 로더 초기화
docs = loader.load()

# 문서의 내용 출력
print(docs[10].page_content[:300])

```

- 셀 출력 (1.5s)

SPRi AI Brief | 2023-12월호

8

코히어, 데이터 투명성 확보를 위한 데이터 출처 탐색기 공개
코히어와 12개 기관이 광범위한 데이터셋에 대한
KEY Contents

데이터 출처 탐색기, 광범위한 데이터셋 정보 제공을 통해 데이터 투명성 향상
AI 기업 코히어(Cohere)가

```
# 메타데이터 출력
```

```
show_metadata(docs)
```

- 셀 출력

```

[metadata]
['producer', 'creator', 'creationdate', 'author', 'moddate', 'pdfversion', 'source']

[examples]
producer      : Hancm PDF 1.3.0.542
creator       : Hwp 2018 10.0.0.13462
creationdate  : 2023-12-08T13:28:38+09:00
author        : dj
moddate       : 2023-12-08T13:28:38+09:00
pdfversion    : 1.4
source        : ../06_Document_Loader/data/SPRI_AI_Brief_2023년12월호_F.pdf
total_pages   : 23
page          : 0
page_label    : 1

```

- 일부 PDF에는 스캔된 문서 or 그림 내에 텍스트 이미지 포함
- `rapidocr-onnxruntime` 패키지 사용 → 이미지에서 텍스트를 추출할 수도 있음
 - 먼저 터미널에 설치
 -

```
pip install -qU rapidocr-onnxruntime
```

```
# PDF 로더 초기화, 이미지 추출 옵션 활성화
loader = PyPDFLoader("https://arxiv.org/pdf/2103.15348.pdf", extract_images=True)

# PDF 페이지 로드
docs = loader.load()

# 페이지 내용 접근
print(docs[4].page_content[:300])
```

- 셀 출력 (0.9s)

```
LayoutParser: A Unified Toolkit for DL-Based DIA 5
Table 1: Current layout detection models in the LayoutParser model zoo
Dataset Base Model1 Large ModelNotes
PubLayNet [38] F / M M Layouts of modern scientific documents
PRImA [3] M – Layouts of scanned modern magazines and scientific reports
Newspaper
```

```
# 메타데이터 출력
```

```
show_metadata(docs)
```

- 셀 출력

```
[metadata]
['producer', 'creator', 'creationdate', 'author', 'keywords', 'moddate', 'ptex.

[examples]
producer      : pdfTeX-1.40.21
creator       : LaTeX with hyperref
creationdate  : 2021-06-22T01:27:10+00:00
author        :
keywords      :
moddate       : 2021-06-22T01:27:10+00:00
ptex.fullbanner : This is pdfTeX, Version 3.14159265-2.6-1.40.21 (TeX Live 2020
subject       :
title         :
trapped       : /False
```

```
source      : https://arxiv.org/pdf/2103.15348.pdf
total_pages : 16
page        : 0
page_label  : 1
```



PyMuPDF

- 속도 최적화
- **PDF** 및 해당 페이지에 대한 자세한 메타데이터를 포함
- **페이지 당 하나의 문서를 반환**
 - 먼저 터미널에 설치
 -

```
pip install -qU pymupdf
```

```
from langchain_community.document_loaders import PyMuPDFLoader

# PyMuPDF 로더 인스턴스 생성
loader = PyMuPDFLoader(FILE_PATH)

# 문서 로드
docs = loader.load()

# 문서의 내용 출력
print(docs[10].page_content[:300])
```

- 셀 출력 (1.0s)

```
SPRi AI Brief |
2023-12월호
8
코히어, 데이터 투명성 확보를 위한 데이터 출처 탐색기 공개
n 코히어와 12개 기관이 광범위한 데이터셋에 대한 감사를 통해 원본 데이터 출처, 재라이선스 상태,
작성자 등 다양한 정보를 제공하는 ‘데이터 출처 탐색기’ 플랫폼을 출시
n 대화형 플랫폼을 통해 개발자는 데이터셋의 라이선스 상태를 쉽게 파악할 수 있으며 데이터셋의
구성과 계보도 추적 가능
KEY Contents
£ 데이터 출처 탐색기, 광범위한 데이터셋 정보 제공을 통해 데이터 투명성 향상
n AI 기업 코히어
```

메타데이터 출력

```
show_metadata(docs)
```

- 셀 출력

```
[metadata]
['producer', 'creator', 'creationdate', 'source', 'file_path', 'total_pages', 'format', 'title', 'author', 'subject', 'keywords', 'moddate', 'trapped', 'modDate', 'creationDate', 'page']

[examples]
producer      : Hancom PDF 1.3.0.542
creator       : Hwp 2018 10.0.0.13462
creationdate  : 2023-12-08T13:28:38+09:00
source        : ../06 Document Loader/data/SPRI AI Brief 2023년12월호_F.pdf
file_path     : ../06 Document Loader/data/SPRI AI Brief 2023년12월호_F.pdf
total_pages   : 23
format        : PDF 1.4
title         :
author        : dj
subject       :
keywords      :
moddate       : 2023-12-08T13:28:38+09:00
trapped       :
modDate       : D:20231208132838+09'00'
creationDate  : D:20231208132838+09'00'
page          : 0
```

▼

Unstructured

- **Unstructured**: Markdown이나 PDF와 같은 비구조화된 또는 반구조화된 파일 형식을 다루기 위한 공통 인터페이스를 지원
- LangChain의 **UnstructuredPDFLoader** = **Unstructured**와 통합 → PDF 문서를 LangChain **Document** 객체로 파싱
 - 먼저 터미널에 설치
 -

```
pip install -qU unstructured
```

```
from langchain_community.document_loaders import UnstructuredPDFLoader
```

```
# UnstructuredPDFLoader 인스턴스 생성 (한국어 설정)
```

```

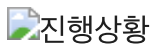
loader = UnstructuredPDFLoader(
    FILE_PATH,
    # languages=["kor"],          # 한국어 지정
    languages=["kor+eng"],        # 한국어+영어 지정
    strategy="hi_res",            # 고해상도 전략 사용
    infer_table_structure=True,   # 표 구조 추론 활성화
    extract_images_in_pdf=False,  # 이미지 추출 비활성화 (문제 발생 시)
    extract_image_block_types=["Figure", "Table"] # 추출할 이미지 유형 지정
)

# 데이터 로드
docs = loader.load()

# 문서의 내용 출력
print(docs[0].page_content[:300])

```

- 셀 출력_1: 실패 (2m 0.1s)



진행상황

The `max_size` parameter is deprecated and will be removed in v4.26. Please spec

S SPR 소 프 특 웨 어 정 책 연구소 S | Software Policy & Research Institute

2023 년 12 월 호

ox 내 zt ~ 1 = |

ono qu m Cc z

HW ro 더 ol El =

~ ob | 로 시

oe HH Pad 6

of oa fe

[요 □□

뜨

N x / 언 브 , rz 0f 브

> 미국 프런티어 모델 포럼, 1,000 만 달러 규 모 의 코 fe} 히어, 데이터 투명성 확 보 를 위한 데이터

두번째 시도

```
from langchain_community.document_loaders import UnstructuredPDFLoader
```

```
loader = UnstructuredPDFLoader(  
    FILE_PATH,  
    languages=["kor+eng"],  
    strategy="hi_res",  
    infer_table_structure=True,  
    extract_images_in_pdf=False,  
    tesseract_path="/opt/homebrew/bin/tesseract",  
    tessdata_path="/opt/homebrew/share/tessdata"  
)
```

```
# 데이터 로드  
docs = loader.load()  
  
# 문서의 내용 출력  
print(docs[0].page_content[:300])
```

- 셀 출력 (경로 계속 인식 못함, 실패) (1m 39.3s)

S SPR 소 프 특 웨 어 정 책 연구소 S | Software Policy & Research Institute

2023 년 12 월 호

ox 내 zt ~ 1 = |

ono qu m Cc z

HW ro 더 ol El =

~ ob | 로 시

oe HH Pad 6

of oa fe

[요 □□

뜨

N x / 언 브 , rz Of 브

> 미국 프린티어 모델 포럼, 1,000 만 달러 규 모 의 코 fe} 히어, 데이터 투명성 확 보 를 위한 데이터

```
# 세번째 시도
```



```

from langchain_community.document_loaders import UnstructuredPDFLoader

loader = UnstructuredPDFLoader(
    FILE_PATH,
    languages=["kor+eng"],
    strategy="hi_res",
    infer_table_structure=True,
    extract_images_in_pdf=False,
    tesseract_path="/opt/homebrew/bin/tesseract",
    #tessdata_path="/opt/homebrew/Cellar/tesseract/5.5.1/share/tessdata"
)

# 데이터 로드
docs = loader.load()

# 문서의 내용 출력
print(docs[0].page_content[:300])

```

- 셀 출력 (실패 3) (1m 39.3s)

S SPR 소 프 특 웨 어 정 책 연구소 S | Software Policy & Research Institute

2023 년 12 월 호

ox 내 zt ~ 1 = |

ono qu m Cc z

HW ro 더 ol El =

~ ob | 로 시

oe HH Pad 6

of oa fe

[요 □□

뜨

N x / 언 브 , rz Of 브

> 미국 프런티어 모델 포럼, 1,000 만 달러 규 모 의 코 fe} 히어, 데이터 투명성 확 보 를 위한 데이터

메타데이터 확인

```
show_metadata(docs)
```

- 셀 출력

```
[metadata]
['source']
```

```
[examples]
```

```
source : ../06 Document Loader/data/SPRI AI Brief 2023년12월호_F.pdf
```

- 내부적으로 비정형에서는 텍스트 청크마다 서로 다른 요소 만들
- 기본적으로 결합되어 있지만 분리 가능 → `mode="elements"` 지정

```
# UnstructuredPDFLoader 인스턴스 생성(mode="elements")
loader = UnstructuredPDFLoader(FILE_PATH, mode="elements")

# 데이터 로드
docs = loader.load()

# 문서의 내용 출력
print(docs[0].page_content)
```

- 셀 출력_1 (실패)(1m 4.1s)

```
Warning: No languages specified, defaulting to English.
S SPR Arete S | Software Policy & Research Institute
```

```
# 두번째 시도_경로와 언어 추가해보기

# UnstructuredPDFLoader 인스턴스 생성(mode="elements")
loader = UnstructuredPDFLoader(
    FILE_PATH,
    languages=["kor+eng"],
    strategy="hi_res",
    infer_table_structure=True,
    extract_images_in_pdf=False,
    tesseract_path="/opt/homebrew/bin/tesseract",
    tessdata_path="/opt/homebrew/share/tessdata",
    mode="elements")

# 데이터 로드
docs = loader.load()

# 문서의 내용 출력
print(docs[0].page_content)
```

- 셀 출력 (O) (2m 43.8s)

SPRI AI Brief

- 이 특정 문서에 대한 전체 요소의 유형 집합을 참조하기

```
# 데이터 카테고리 추출
```

```
set(doc.metadata["category"] for doc in docs)
```

- 셀 출력

```
{'FigureCaption',  
'Header',  
'Image',  
'ListItem',  
'NarrativeText',  
'Table',  
'Title',  
'UncategorizedText'}
```

```
# 메타데이터 확인
```

```
show_metadata(docs)
```

- 셀 출력

```
[metadata]  
['source', 'detection_class_prob', 'coordinates', 'last_modified', 'filetype',  
  
[examples]  
source           : ../06 Document Loader/data/SPRI AI Brief 2023년12월호_F.pdf  
detection_class_prob : 0.42657050490379333  
coordinates       : {'points': ((np.float64(245.91224670410156), np.float64(4  
last_modified     : 2025-09-12T14:31:21  
filetype          : application/pdf  
languages         : ['kor']  
page_number       : 1  
file_directory    : ../06 Document Loader/data  
filename          : SPRI_AI_Brief_2023년12월호_F.pdf  
category          : Title  
element_id        : 9052c04dc037753bb06efd7ab0a37bf2
```

- 대안

```
import pdfplumber

with pdfplumber.open(FILE_PATH) as pdf:
    text = ""
    for page in pdf.pages:
        text += page.extract_text()
    print(text[:300])
```

- 셀 출력 (1.9s)

```
12
2023년 월호2023년 12월호
|
. 인공지능 산업 동향 브리프
1. 정책/법제
> 미국, 안전하고 신뢰할 수 있는 AI 개발과 사용에 관한 행정명령 발표 .....1
> G7, 히로시마 AI 프로세스를 통해 AI 기업 대상 국제 행동강령에 합의.....
> 영국 AI 안전성 정상회의에 참가한 28개국, AI 위험에 공동 대응 선언.....
> 미국 법원, 예술가들이 생성 AI 기업에 제기한 저작권 소송 기
```

▼

PyPDFium2

- **PyPDFium2** = PDF 파일을 처리하는 데 사용되는 라이브러리 / **더 빠르고 안정적**
- `get_text_range()` 와 `get_text_bounded()` 는 모두 텍스트를 추출하는 함수이지만, 매개변수와 동작 방식이 다를 수 있음

```
from langchain_community.document_loaders import PyPDFium2Loader

# PyPDFium2 로더 인스턴스 생성
loader = PyPDFium2Loader(FILE_PATH)

# 데이터 로드
docs = loader.load()

# 문서의 내용 출력
print(docs[10].page_content[:300]) # 10번째 페이지의 처음 300자만 출력하도록
```

- 셀 출력 (0.5s)

코히어, 데이터 투명성 확보를 위한 데이터 출처 탐색기 공개

n 코히어와 12개 기관이 광범위한 데이터셋에 대한 감사를 통해 원본 데이터 출처, 재라이선스 상태, 작성자 등
n 대화형 플랫폼을 통해 개발자는 데이터셋의 라이선스 상태를 쉽게 파악할 수 있으며 데이터셋의
구성과 계보도 추적 가능

KEY Contents

£ 데이터 출처 탐색기, 광범위한 데이터셋 정보 제공을 통해 데이터 투명성 향상

n AI 기업 코히어(Co

• 경고 메시지 해석

[/Users/jay/.pyenv/versions/lc env/lib/python3.13/site-packages/pypdfium2/ help](#)
warnings.warn("get_text_range() call with default params will be implicitly red

- PyPDFium2 라이브러리에서 `get_text_range()` 함수가 기본 매개변수로 호출되었을 때, 내
부적으로 `get_text_bounded()` 함수로 리디렉션(재지정)된다는 경고 → 무시해도 됨
 - 이 경고는 코드의 동작에는 영향을 주지 않지만, 향후 버전에서 `get_text_range()` 함수의
기본 동작이 변경될 수 있음을 알려줌
 - 현재는 `get_text_bounded()` 함수를 사용하여 텍스트 추출

메타데이터 출력

`show_metadata(docs)`

• 셀 출력

```
[metadata]
['producer', 'creator', 'creationdate', 'title', 'author', 'subject', 'keywords']

[examples]
producer      : Hancm PDF 1.3.0.542
creator       : Hwp 2018 10.0.0.13462
creationdate  : 2023-12-08T13:28:38+09:00
title         :
author        : dj
subject       :
keywords      :
moddate       : 2023-12-08T13:28:38+09:00
source        : ../06_Document Loader/data/SPRI_AI_Brief_2023년12월호_F.pdf
total_pages   : 23
page          : 0
```



PDFMiner

```
from langchain_community.document_loaders import PDFMinerLoader

# PDFMiner 로더 인스턴스 생성
loader = PDFMinerLoader(FILE_PATH)

# 데이터 로드
docs = loader.load()

# 문서의 내용 출력
print(docs[0].page_content[:300])
```

- 셀 출력 (1.7s)

2023년 12월호

2023년 12월호

I. 인공지능 산업 동향 브리프

1. 정책/법제

- ▶ 미국, 안전하고 신뢰할 수 있는 AI 개발과 사용에 관한 행정명령 발표
- ▶ G7, 히로시마 AI 프로세스를 통해 AI 기업 대상 국제 행동강령에 합의
- ▶ 영국 AI 안전성 정상회의에 참가한 28개국, AI 위험에 공동

```
# 메타데이터 출력
```

```
show_metadata(docs)
```

- 셀 출력

```
[metadata]
['producer', 'creator', 'creationdate', 'author', 'moddate', 'pdfversion', 'totalpages']

[examples]
producer      : Hancom PDF 1.3.0.542
creator       : Hwp 2018 10.0.0.13462
creationdate  : 2023-12-08T13:28:38+09:00
author        : dj
```

```
moddate      : 2023-12-08T13:28:38+09:00
pdfversion   : 1.4
total_pages  : 23
source       : ../06 Document Loader/data/SPRI AI Brief 2023년12월호_F.pdf
```

- **PDFMiner** 를 사용하여 **HTML 텍스트** 생성
 - 이 방법은 출력된 **HTML 콘텐츠** → **BeautifulSoup** 을 통해 파싱
 - **글꼴 크기**, **페이지 번호**, **PDF 헤더/푸터** 등에 대한 보다 구조화되고 풍부한 정보를 얻을 수 있음
→ 텍스트를 의미론적으로 섹션으로 분할하는 데 도움이 될 수 있음

```
# PDFMinerPDFasHTMLLoader를 사용하여 PDF 문서를 HTML 형식으로 로드하는 코드

# langchain_community.document_loaders 모듈에서 PDFMinerPDFasHTMLLoader 클래스 импорт
from langchain_community.document_loaders import PDFMinerPDFasHTMLLoader

# PDFMinerPDFasHTMLLoader 인스턴스 생성
# FILE_PATH: 로드할 PDF 파일의 경로
loader = PDFMinerPDFasHTMLLoader(FILE_PATH)

# 문서 로드
# loader.load(): PDF 파일을 HTML 형식으로 로드하여 Document 객체 리스트 반환
docs = loader.load()

# 문서의 내용 출력
# docs[0].page_content: 첫 번째 페이지의 내용
# [:300]: 내용의 처음 300자만 출력
print(docs[0].page_content[:300])
```

- 셀 출력 (1.7s)

```
<html><head>
<meta http-equiv="Content-Type" content="text/html">
</head><body>
<span style="position:absolute; border: gray 1px solid; left:0px; top:50px; wid
<div style="position:absolute; top:50px;"><a name="1">Page 1</a></div>
<div style="position:absolute; border
```

```
# 메타데이터 출력

show_metadata(docs)
```

- 셀 출력

[metadata]

['source']

[examples]

source : [../06 Document Loader/data/SPRI AI Brief 2023년12월호_F.pdf](#)

```
from bs4 import BeautifulSoup
```

```
# BeautifulSoup 객체 생성
```

```
# docs[0].page_content: HTML 형식의 문자열
```

```
# "html.parser": HTML 파서 지정
```

```
soup = BeautifulSoup(docs[0].page_content, "html.parser")
```

```
# 모든 div 태그 검색
```

```
# soup.find_all("div"): HTML 문서에서 모든 div 태그를 찾아 리스트로 반환
```

```
content = soup.find_all("div")
```

- 이 코드는 `BeautifulSoup` 라이브러리를 사용 → `HTML` 문서 파싱 → 모든 `div` 태그를 검색하는 기능 수행
- `BeautifulSoup` 객체 생성 → `find_all` 메서드를 사용 → 모든 `div` 태그를 찾아 `리스트`로 반환 → `content` 변수에 저장

```
import re
```

```
# 정규표현식을 사용하기 위한 파이썬 임포트
```

```
# 현재 글꼴 크기 초기화
```

```
cur_fs = None
```

```
# 현재 텍스트 초기화
```

```
cur_text = ""
```

```
# 동일한 글꼴 크기의 모든 스니펫 수집할 리스트
```

```
snippets = []
```

```
# 모든 div 태그를 순회
```

```
for c in content:
```

```
    sp = c.find("span")
```

```
# div 태그 내의 span 태그 검색
```

```
    if not sp:
```

```
# span 태그가 없으면 다음 태그로 넘어감
```

```
        continue
```

```
# span 태그의 style 속성 가져오기
```

```
st = sp.get("style")
```

```
if not st:
```

```
# style 속성이 없으면 다음 태그로 넘어감
```

```
    continue
```

```
# style 속성에서 font-size 값 추출
```

```
#fs = re.findall("font-size:(\d+)px", st)
```

```
# 수정된 코드
```

```
fs = re.findall(r"font-size:(\d+)px", st)
```

```
if not fs:
```

```
# font-size 값이 없으면 다음 태그로 넘어감
```



```

        continue

    # font-size 값을 정수로 변환
    fs = int(fs[0])

    # 현재 글꼴 크기가 설정되지 않았으면 현재 글꼴 크기로 설정
    if not cur_fs:
        cur_fs = fs

    # 현재 글꼴 크기와 동일한 경우 텍스트 추가
    if fs == cur_fs:
        cur_text += c.text
    # 글꼴 크기가 다른 경우
    else:
        # 현재까지의 텍스트와 글꼴 크기를 snippets 리스트에 추가
        snippets.append((cur_text, cur_fs))
        # 현재 글꼴 크기와 텍스트 업데이트
        cur_fs = fs
        cur_text = c.text

# 마지막 텍스트와 글꼴 크기를 snippets 리스트에 추가
snippets.append((cur_text, cur_fs))

# 중복 스니펫 제거 전략 추가 가능성 (PDF의 헤더/푸터가 여러 페이지에 걸쳐 나타나므로 중복 발견 시 중복 :

```

- **HTML** 문서에서 **동일한 글꼴 크기** 를 가진 **텍스트 스니펫** 을 **추출** 하는 기능을 수행
- **BeautifulSoup** 을 사용하여 **HTML 문서** 를 **파싱** 한 후, 각 **div** 태그 내의 **span** 태그에서 **글꼴 크기** 를 추출하고, **동일한 글꼴 크기** 를 가진 **텍스트** 를 모아 **리스트** 에 저장
 - 이 리스트는 **snippets** 변수에 저장되며
 - 각 요소는 (**텍스트, 글꼴 크기**)의 **튜플**

```

from langchain_core.documents import Document

# 현재 인덱스 초기화
cur_idx = -1
# 의미론적 스니펫 리스트 초기화
semantic_snippets = []

# 제목 가정: 높은 글꼴 크기
for s in snippets:
    # 새 제목 판별: 현재 스니펫 글꼴 > 이전 제목 글꼴
    if (
        not semantic_snippets          # semantic_snippets가 비어있는 경우
        or s[1] > semantic_snippets[cur_idx].metadata["heading_font"]
                                         # 현재 글꼴 크기가 이전 제목 글꼴 크기보다 큰 경우
    ):
        # 새 문서 생성
        metadata = {
            "heading": s[0],          # 제목으로 현재 텍스트 설정
            "content_font": 0,        # 내용 글꼴 초기화
            "heading_font": s[1]      # 제목 글꼴 크기 설정
        }

```

```

    }
    metadata.update(docs[0].metadata)          # 기존 메타데이터 추가
    semantic_snippets.append(Document(page_content="", metadata=metadata))
    cur_idx += 1                               # 인덱스 증가
    continue

# 동일 섹션 내용 판별: 현재 스니펫 글꼴 <= 이전 내용 글꼴
if (
    not semantic_snippets[cur_idx].metadata["content_font"]
        # 내용 글꼴이 설정되지 않은 경우
    or s[1] <= semantic_snippets[cur_idx].metadata["content_font"]
        # 현재 글꼴 크기가 이전 내용 글꼴 크기보다 작거나 같은 경우
):
    # 현재 섹션에 내용 추가
    semantic_snippets[cur_idx].page_content += s[0]          # 텍스트 추가
    semantic_snippets[cur_idx].metadata["content_font"] = max(
        s[1], semantic_snippets[cur_idx].metadata["content_font"]  # 더 큰 글꼴 크기
    )
    continue

# 새 섹션 생성 조건: 현재 스니펫 글꼴 > 이전 내용 글꼴, 이전 제목 글꼴 미만
metadata = {
    "heading": s[0],          # 제목으로 현재 텍스트 설정
    "content_font": 0,        # 내용 글꼴 초기화
    "heading_font": s[1]      # 제목 글꼴 크기 설정
}
metadata.update(docs[0].metadata)          # 기존 메타데이터 추가
semantic_snippets.append(Document(page_content="", metadata=metadata))
cur_idx += 1                               # 인덱스 증가

# 5번째 의미론적 스니펫 출력
print(semantic_snippets[4])

```

- **PDF** 문서의 구조를 분석 → **의미론적 스니펫**을 **생성**하는 기능을 수행
- 글꼴 크기를 기반으로 **제목**과 **내용**을 구분하고, 각 **섹션**을 문서 객체로 생성
- **semantic_snippets** 리스트 = 각 **섹션**의 **제목**, **내용**, **글꼴 크기** 정보가 포함된 **문서 객체** 저장

- 셀 출력

page_content='KEY Contents

n 미국 바이든 대통령이 ‘안전하고 신뢰할 수 있는 AI 개발과 사용에 관한 행정명령’에 서명하고 광범위한 행정 조치를 명시

n 행정명령은 △AI의 안전과 보안 기준 마련 △개인정보보호 △형평성과 시민권 향상 △소비자 보호 △노동자 지원 △혁신과 경쟁 촉진 △국제협력을 골자로 함

```
' metadata={'heading': '미국, 안전하고 신뢰할 수 있는 AI 개발과 사용에 관한 행정명령
```



PyPDF 디렉토리

- 디렉토리에서 PDF 로드하기

```
from langchain_community.document_loaders import PyPDFDirectoryLoader

# 디렉토리 경로
loader = PyPDFDirectoryLoader("../06_Document_Loader/data/")

# 문서 로드
docs = loader.load()

# 문서의 개수 출력
print(len(docs))
```

```
# 44
```

```
44
```

```
print(type(docs))
```

```
# <class 'list'>
```

```
print(docs)
```

- 셀 출력

```
[Document(metadata={'producer': 'Call PDF v 2.4', 'creator': 'Call PDF', 'creat
```

```
# metadata 출력
```

```
print(docs[1].metadata)
```

- 셀 출력

```
{'producer': 'Call PDF v 2.4', 'creator': 'Call PDF', 'creationdate': '', 'titl
```

```
# 문서의 내용 출력
```

```
print(docs[3].page_content[:300])
```

- 셀 출력

II. 디지털 정부혁신 추진계획▶ (비전) 디지털로 여는 좋은 세상 * 부제 : 대한민국이 먼저 갑니다.▶ (추진
DB화하고, 한번에안내

추천

신청

결과확인까지가능한통합서비스환경구현 * PC, 스마트폰, AI스피커 등 다양한 기기에서 인공지능 기반의 처

▽

PDFlumber

- **PyMuPDF** 와 유사
- 출력: **PDF** 와 그 페이지에 대한 자세한 **메타데이터** 를 포함 + **페이지 당 하나의 문서를 반환**

```
from langchain_community.document_loaders import PDFPlumberLoader

# PDF 문서 로더 인스턴스 생성
loader = PDFPlumberLoader(FILE_PATH)

# 문서 로딩
docs = loader.load()

# 첫 번째 문서 데이터 접근
print(docs[10].page_content[:300])
```

- 셀 출력 (2.0s)

```
SPRi AI Brief |
2023-12월호
코히어, 데이터 투명성 확보를 위한 데이터 출처 탐색기 공개
KEY Contents
n 코히어와 12개 기관이 광범위한 데이터셋에 대한 감사를 통해 원본 데이터 출처, 재라이선스 상태,
작성자 등 다양한 정보를 제공하는 ‘데이터 출처 탐색기’ 플랫폼을 출시
n 대화형 플랫폼을 통해 개발자는 데이터셋의 라이선스 상태를 쉽게 파악할 수 있으며 데이터셋의
구성과 계보도 추적 가능
£데이터 출처 탐색기, 광범위한 데이터셋 정보 제공을 통해 데이터 투명성 향상
n AI 기업 코히어(Cohere)
```

```
# 메타데이터 출력
```

```
show_metadata(docs)
```

- 셀 출력

```
[metadata]
['source', 'file_path', 'page', 'total_pages', 'Author', 'Creator', 'Producer',

[examples]
source      : ../06 Document Loader/data/SPRI AI Brief 2023년12월호_F.pdf
file_path   : ../06 Document Loader/data/SPRI AI Brief 2023년12월호_F.pdf
page        : 0
total_pages : 23
Author      : dj
Creator     : Hwp 2018 10.0.0.13462
Producer    : Hancom PDF 1.3.0.542
CreationDate : D:20231208132838+09'00'
ModDate     : D:20231208132838+09'00'
PDFVersion  : 1.4
```

-
- *next:* 한글(**HWP**)
-