Unstructured File

This notebook covers how to use `Unstructured` package to load files of many types. `Unstructured` currently supports loading of text files, powerpoints, html, pdfs, images, and more.

```
# # Install package
pip install "unstructured[all-docs]"
```

```
# # Install other dependencies
# # https://github.com/Unstructured-
IO/unstructured/blob/main/docs/source/installing.rst
# !brew install libmagic
# !brew install poppler
# !brew install tesseract
# # If parsing xml / html documents:
# !brew install libxml2
# !brew install libxslt
```

```
# import nltk
# nltk.download('punkt')
```

```
from langchain.document_loaders import UnstructuredFileLoader
```

API Reference:

- UnstructuredFileLoader

```
loader = UnstructuredFileLoader("./example_data/state_of_the_union.txt")
```

```
docs = loader.load()
```

```
docs[0].page_content[:400]
```

```
    'Madam Speaker, Madam Vice President, our First Lady and Second Gentleman. Members of Congress and the Cabinet. Justices of the Supreme Court. My fellow Americans.\n\nLast year COVID-19 kept us apart. This year we are finally together again.\n\nTonight, we meet as Democrats Republicans and Independents. But most importantly as Americans.\n\nWith a duty to one another to the American people to the Constit'
```

Retain Elements

Under the hood, Unstructured creates different "elements" for different chunks of text. By default we combine those together, but you can easily keep that separation by specifying `mode="elements"`.

```
loader = UnstructuredFileLoader(
    "./example_data/state_of_the_union.txt", mode="elements"
)
```

```
docs = loader.load()
```

```
docs[:5]
```

```
    [Document(page_content='Madam Speaker, Madam Vice President, our First Lady and Second Gentleman. Members of Congress and the Cabinet. Justices of the Supreme Court. My fellow Americans.', lookup_str='', metadata={'source': '../../state_of_the_union.txt'}, lookup_index=0),
     Document(page_content='Last year COVID-19 kept us apart. This year we are finally together again.', lookup_str='', metadata={'source': '../../state_of_the_union.txt'}, lookup_index=0),
     Document(page_content='Tonight, we meet as Democrats Republicans and Independents. But most importantly as Americans.', lookup_str='', metadata={'source': '../../state_of_the_union.txt'}, lookup_index=0),
     Document(page_content='With a duty to one another to the American people to the Constitution.', lookup_str='', metadata={'source': '../../state_of_the_union.txt'}, lookup_index=0),
     Document(page_content='And with an unwavering resolve that freedom will always triumph over tyranny.', lookup_str='', metadata={'source': '../../state_of_the_union.txt'}, lookup_index=0)]
```

Define a Partitioning Strategy

Unstructured document loader allow users to pass in a `strategy` parameter that lets `unstructured` know how to partition the document. Currently supported strategies are `"hi_res"` (the default) and `"fast"`. Hi res partitioning strategies are more accurate, but take longer to process. Fast strategies partition the document more quickly, but trade-off accuracy. Not all document types have separate hi res and fast partitioning strategies. For those document types, the `strategy` kwarg is ignored. In some cases, the high res strategy will fallback to fast if there is a dependency missing (i.e. a model for document partitioning). You can see how to apply a strategy to an `UnstructuredFileLoader` below.

```
from langchain.document_loaders import UnstructuredFileLoader
```

API Reference:

- UnstructuredFileLoader

```
loader = UnstructuredFileLoader(
    "layout-parser-paper-fast.pdf", strategy="fast", mode="elements"
)
```

```
docs = loader.load()
```

```
docs[:5]
```

```
    [Document(page_content='1', lookup_str='', metadata={'source': 'layout-parser-paper-fast.pdf', 'filename': 'layout-parser-paper-fast.pdf', 'page_number': 1, 'category': 'UncategorizedText'}, lookup_index=0),
     Document(page_content='2', lookup_str='', metadata={'source': 'layout-parser-paper-fast.pdf', 'filename': 'layout-parser-paper-fast.pdf', 'page_number': 1, 'category': 'UncategorizedText'}, lookup_index=0),
     Document(page_content='0', lookup_str='', metadata={'source': 'layout-parser-paper-fast.pdf', 'filename': 'layout-parser-paper-fast.pdf', 'page_number': 1, 'category': 'UncategorizedText'}, lookup_index=0),
     Document(page_content='2', lookup_str='', metadata={'source': 'layout-parser-paper-fast.pdf', 'filename': 'layout-parser-paper-fast.pdf', 'page_number': 1, 'category': 'UncategorizedText'}, lookup_index=0),
     Document(page_content='n', lookup_str='', metadata={'source': 'layout-
parser-paper-fast.pdf', 'filename': 'layout-parser-paper-fast.pdf',
'page_number': 1, 'category': 'Title'}, lookup_index=0)]
```

PDF Example

Processing PDF documents works exactly the same way. Unstructured detects the file type and extracts the same types of elements. Modes of operation are

- `single` all the text from all elements are combined into one (default)
- `elements` maintain individual elements
- `paged` texts from each page are only combined

```
wget  https://raw.githubusercontent.com/Unstructured-
IO/unstructured/main/example-docs/layout-parser-paper.pdf -P "../../"
```

```
loader = UnstructuredFileLoader(
    "./example_data/layout-parser-paper.pdf", mode="elements"
)
```

```
docs = loader.load()
```

```
docs[:5]
```

```
    [Document(page_content='LayoutParser : A Unified Toolkit for Deep Learning
Based Document Image Analysis', lookup_str='', metadata={'source':
'../../layout-parser-paper.pdf'}, lookup_index=0),
     Document(page_content='Zejiang Shen 1 ( (ea)\n ), Ruochen Zhang 2 ,
Melissa Dell 3 , Benjamin Charles Germain Lee 4 , Jacob Carlson 3 , and
Weining Li 5', lookup_str='', metadata={'source': '../../layout-parser-
paper.pdf'}, lookup_index=0),
     Document(page_content='Allen Institute for AI shannons@allenai.org',
lookup_str='', metadata={'source': '../../layout-parser-paper.pdf'},
lookup_index=0),
     Document(page_content='Brown University ruochen zhang@brown.edu',
lookup_str='', metadata={'source': '../../layout-parser-paper.pdf'},
lookup_index=0),
     Document(page_content='Harvard University { melissadell,jacob carlson }
@fas.harvard.edu', lookup_str='', metadata={'source': '../../layout-parser-
paper.pdf'}, lookup_index=0)]
```

If you need to post process the `unstructured` elements after extraction, you can pass in a list of `str` -> `str` functions to the `post_processors` kwarg when you instantiate the `UnstructuredFileLoader`. This applies to other Unstructured loaders as well. Below is an example.

```
from langchain.document_loaders import UnstructuredFileLoader
from unstructured.cleaners.core import clean_extra_whitespace
```

API Reference:

- UnstructuredFileLoader

```
loader = UnstructuredFileLoader(
    "./example_data/layout-parser-paper.pdf",
    mode="elements",
    post_processors=[clean_extra_whitespace],
)
```

```
docs = loader.load()
```

```
docs[:5]
```

```
    [Document(page_content='LayoutParser: A Unified Toolkit for Deep Learning
Based Document Image Analysis', metadata={'source': './example_data/layout-
parser-paper.pdf', 'coordinates': {'points': ((157.62199999999999,
114.23496279999995), (157.62199999999999, 146.5141628), (457.7358962799999,
146.5141628), (457.7358962799999, 114.23496279999995)), 'system':
'PixelSpace', 'layout_width': 612, 'layout_height': 792}, 'filename': 'layout-
parser-paper.pdf', 'file_directory': './example_data', 'filetype':
'application/pdf', 'page_number': 1, 'category': 'Title'}),
     Document(page_content='Zejiang Shen1 ((cid:0)), Ruochen Zhang2, Melissa
Dell3, Benjamin Charles Germain Lee4, Jacob Carlson3, and Weining Li5',
metadata={'source': './example_data/layout-parser-paper.pdf', 'coordinates':
{'points': ((134.809, 168.64029940800003), (134.809, 192.2517444),
(480.5464199080001, 192.2517444), (480.5464199080001, 168.64029940800003)),
'system': 'PixelSpace', 'layout_width': 612, 'layout_height': 792},
'filename': 'layout-parser-paper.pdf', 'file_directory': './example_data',
'filetype': 'application/pdf', 'page_number': 1, 'category':
'UncategorizedText'}),
     Document(page_content='1 Allen Institute for AI shannons@allenai.org 2
Brown University ruochen zhang@brown.edu 3 Harvard University
{melissadell,jacob carlson}@fas.harvard.edu 4 University of Washington
bcgl@cs.washington.edu 5 University of Waterloo w422li@uwaterloo.ca',
metadata={'source': './example_data/layout-parser-paper.pdf', 'coordinates':
{'points': ((207.23000000000002, 202.57205439999996), (207.23000000000002,
311.8195408), (408.12676, 311.8195408), (408.12676, 202.57205439999996)),
'system': 'PixelSpace', 'layout_width': 612, 'layout_height': 792},
'filename': 'layout-parser-paper.pdf', 'file_directory': './example_data',
'filetype': 'application/pdf', 'page_number': 1, 'category':
'UncategorizedText'}),
     Document(page_content='1 2 0 2', metadata={'source':
'./example_data/layout-parser-paper.pdf', 'coordinates': {'points': ((16.34,
213.36), (16.34, 253.36), (36.34, 253.36), (36.34, 213.36)), 'system':
'PixelSpace', 'layout_width': 612, 'layout_height': 792}, 'filename': 'layout-
parser-paper.pdf', 'file_directory': './example_data', 'filetype':
'application/pdf', 'page_number': 1, 'category': 'UncategorizedText'}),
     Document(page_content='n u J', metadata={'source':
'./example_data/layout-parser-paper.pdf', 'coordinates': {'points': ((16.34,
258.36), (16.34, 286.14), (36.34, 286.14), (36.34, 258.36)), 'system':
'PixelSpace', 'layout_width': 612, 'layout_height': 792}, 'filename': 'layout-
parser-paper.pdf', 'file_directory': './example_data', 'filetype':
'application/pdf', 'page_number': 1, 'category': 'Title'})]
```

Unstructured API

If you want to get up and running with less set up, you can simply run `pip install unstructured` and use `UnstructuredAPIFileLoader` or `UnstructuredAPIFileIOLoader`. That will process your document using the hosted Unstructured API. You can generate a free Unstructured API key here. The Unstructured documentation page will have instructions on how to generate an API key once they're available. Check out the instructions here if you'd like to self-host the Unstructured API or run it locally.

```python
from langchain.document_loaders import UnstructuredAPIFileLoader
```

API Reference:

- UnstructuredAPIFileLoader

```
filenames = ["example_data/fake.docx", "example_data/fake-email.eml"]
```

```
loader = UnstructuredAPIFileLoader(
    file_path=filenames[0],
    api_key="FAKE_API_KEY",
)
```

```
docs = loader.load()
docs[0]
```

```
    Document(page_content='Lorem ipsum dolor sit amet.', metadata={'source':
'example_data/fake.docx'})
```

You can also batch multiple files through the Unstructured API in a single API using `UnstructuredAPIFileLoader`.

```
loader = UnstructuredAPIFileLoader(
    file_path=filenames,
    api_key="FAKE_API_KEY",
)
```

```
docs = loader.load()
docs[0]
```

```
    Document(page_content='Lorem ipsum dolor sit amet.\n\nThis is a test email
to use for unit tests.\n\nImportant points:\n\nRoses are red\n\nViolets are
blue', metadata={'source': ['example_data/fake.docx', 'example_data/fake-
email.eml']})
```