JSON

> JSON (JavaScript Object Notation) is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and arrays (or other serializable values).

> JSON Lines is a file format where each line is a valid JSON value.

> The `JSONLoader` uses a specified jq schema to parse the JSON files. It uses the `jq` python package. Check this manual for a detailed documentation of the `jq` syntax.

```
#!pip install jq
```

```
from langchain_community.document_loaders
import JSONLoader
import json
from pathlib import Path
from pprint import pprint


file_path='./example_data/facebook_chat.json'
data =
json.loads(Path(file_path).read_text())
```

```
pprint(data)
```

```
    {'image': {'creation_timestamp':
1675549016, 'uri': 'image_of_the_chat.jpg'},
     'is_still_participant': True,
     'joinable_mode': {'link': '', 'mode':
1},
     'magic_words': [],
     'messages': [{'content': 'Bye!',
                   'sender_name': 'User 2',
                   'timestamp_ms':
1675597571851},
                  {'content': 'Oh no worries!
Bye',
                   'sender_name': 'User 1',
                   'timestamp_ms':
1675597435669},
                  {'content': 'No Im sorry it
was my mistake, the blue one is not '
```

```
                              'for sale',
                   'sender_name': 'User 2',
                   'timestamp_ms':
1675596277579},
                  {'content': 'I thought you
were selling the blue one!',
                   'sender_name': 'User 1',
                   'timestamp_ms':
1675595140251},
                  {'content': 'Im not
interested in this bag. Im interested in the
'
                              'blue one!',
                   'sender_name': 'User 1',
                   'timestamp_ms':
1675595109305},
                  {'content': 'Here is $129',
                   'sender_name': 'User 2',
                   'timestamp_ms':
1675595068468},
                  {'photos':
[{'creation_timestamp': 1675595059,
                               'uri':
'url_of_some_picture.jpg'}],
                   'sender_name': 'User 2',
                   'timestamp_ms':
1675595060730},
                  {'content': 'Online is at
least $100',
                   'sender_name': 'User 2',
                   'timestamp_ms':
```

```
1675595045152},
                  {'content': 'How much do
you want?',
                   'sender_name': 'User 1',
                   'timestamp_ms':
1675594799696},
                  {'content': 'Goodmorning!
$50 is too low.',
                   'sender_name': 'User 2',
                   'timestamp_ms':
1675577876645},
                  {'content': 'Hi! Im
interested in your bag. Im offering $50. Let
'
                              'me know if you
are interested. Thanks!',
                   'sender_name': 'User 1',
                   'timestamp_ms':
1675549022673}],
     'participants': [{'name': 'User 1'},
{'name': 'User 2'}],
     'thread_path': 'inbox/User 1 and User 2
chat',
     'title': 'User 1 and User 2 chat'}
```

Using `JSONLoader`

Suppose we are interested in extracting the values under the `content` field within the `messages` key of the JSON data. This can easily be done through the `JSONLoader` as shown below.

JSON file

```
loader = JSONLoader(

file_path='./example_data/facebook_chat.json',
    jq_schema='.messages[].content',
    text_content=False)

data = loader.load()
```

```
pprint(data)
```

```
    [Document(page_content='Bye!', metadata={'s
'/Users/avsolatorio/WBG/langchain/docs/modules/
'seq_num': 1}),
     Document(page_content='Oh no worries! Bye'
'/Users/avsolatorio/WBG/langchain/docs/modules/
'seq_num': 2}),
     Document(page_content='No Im sorry it was 
'/Users/avsolatorio/WBG/langchain/docs/modules/
'seq_num': 3}),
     Document(page_content='I thought you were 
'/Users/avsolatorio/WBG/langchain/docs/modules/
'seq_num': 4}),
     Document(page_content='Im not interested i
'/Users/avsolatorio/WBG/langchain/docs/modules/
'seq_num': 5}),
     Document(page_content='Here is $129', meta
'/Users/avsolatorio/WBG/langchain/docs/modules/
'seq_num': 6}),
     Document(page_content='', metadata={'sourc
'/Users/avsolatorio/WBG/langchain/docs/modules/
'seq_num': 7}),
     Document(page_content='Online is at least
'/Users/avsolatorio/WBG/langchain/docs/modules/
'seq_num': 8}),
     Document(page_content='How much do you wan
'/Users/avsolatorio/WBG/langchain/docs/modules/
'seq_num': 9}),
     Document(page_content='Goodmorning! $50 is
'/Users/avsolatorio/WBG/langchain/docs/modules/
'seq_num': 10}),
     Document(page_content='Hi! Im interested i
Thanks!', metadata={'source':
'/Users/avsolatorio/WBG/langchain/docs/modules/
'seq_num': 11})]
```

JSON Lines file

If you want to load documents from a JSON Lines file, you pass `json_lines=True` and specify `jq_schema` to extract `page_content` from a single JSON object.

```
file_path =
'./example_data/facebook_chat_messages.jsonl'
pprint(Path(file_path).read_text())
```

```
    ('{"sender_name": "User 2",
"timestamp_ms": 1675597571851, "content":
"Bye!"}\n'
     '{"sender_name": "User 1",
"timestamp_ms": 1675597435669, "content": "Oh
no '
     'worries! Bye"}\n'
     '{"sender_name": "User 2",
"timestamp_ms": 1675596277579, "content": "No
Im '
     'sorry it was my mistake, the blue one
is not for sale"}\n')
```

```
loader = JSONLoader(

file_path='./example_data/facebook_chat_message
    jq_schema='.content',
    text_content=False,
    json_lines=True)

data = loader.load()
```

```
pprint(data)
```

```
    [Document(page_content='Bye!', metadata={'s
'langchain/docs/modules/indexes/document_loader
'seq_num': 1}),
     Document(page_content='Oh no worries! Bye'
'langchain/docs/modules/indexes/document_loader
'seq_num': 2}),
     Document(page_content='No Im sorry it was 
{'source':
'langchain/docs/modules/indexes/document_loader
'seq_num': 3})]
```

Another option is set `jq_schema='.'` and provide `content_key`:

```
loader = JSONLoader(

file_path='./example_data/facebook_chat_message
    jq_schema='.',
    content_key='sender_name',
    json_lines=True)

data = loader.load()
```

```
pprint(data)
```

```
    [Document(page_content='User 2', metadata={
'langchain/docs/modules/indexes/document_loader
'seq_num': 1}),
     Document(page_content='User 1', metadata={
'langchain/docs/modules/indexes/document_loader
'seq_num': 2}),
     Document(page_content='User 2', metadata={
'langchain/docs/modules/indexes/document_loader
'seq_num': 3})]
```

Extracting metadata

Generally, we want to include metadata available in the JSON file into the documents that we create from the content.

The following demonstrates how metadata can be extracted using the `JSONLoader`.

There are some key changes to be noted. In the previous example where we didn't collect the metadata, we managed to directly specify in the schema where the value for the `page_content` can be extracted from.

```
.messages[].content
```

In the current example, we have to tell the loader to iterate over the records in the `messages` field. The jq_schema then has to be:

```
.messages[]
```

This allows us to pass the records (dict) into the `metadata_func` that has to be implemented. The `metadata_func` is responsible for identifying which pieces of information in the record should be included in the metadata stored in the final `Document` object.

Additionally, we now have to explicitly specify in the loader, via the `content_key` argument, the key from the record where the value for the `page_content` needs to be extracted from.

```
# Define the metadata extraction function.
def metadata_func(record: dict, metadata:
dict) -> dict:

    metadata["sender_name"] =
record.get("sender_name")
    metadata["timestamp_ms"] =
record.get("timestamp_ms")

    return metadata


loader = JSONLoader(

file_path='./example_data/facebook_chat.json',
    jq_schema='.messages[]',
    content_key="content",
    metadata_func=metadata_func
)

data = loader.load()
```

```
pprint(data)
```

```
    [Document(page_content='Bye!', metadata={'s
'/Users/avsolatorio/WBG/langchain/docs/modules/
'seq_num': 1, 'sender_name': 'User 2', 'timesta
     Document(page_content='Oh no worries! Bye'
'/Users/avsolatorio/WBG/langchain/docs/modules/
'seq_num': 2, 'sender_name': 'User 1', 'timesta
     Document(page_content='No Im sorry it was 
'/Users/avsolatorio/WBG/langchain/docs/modules/
'seq_num': 3, 'sender_name': 'User 2', 'timesta
     Document(page_content='I thought you were 
'/Users/avsolatorio/WBG/langchain/docs/modules/
```

```
 'seq_num': 4, 'sender_name': 'User 1', 'timesta
      Document(page_content='Im not interested i
 '/Users/avsolatorio/WBG/langchain/docs/modules/
 'seq_num': 5, 'sender_name': 'User 1', 'timesta
      Document(page_content='Here is $129', meta
 '/Users/avsolatorio/WBG/langchain/docs/modules/
 'seq_num': 6, 'sender_name': 'User 2', 'timesta
      Document(page_content='', metadata={'sourc
 '/Users/avsolatorio/WBG/langchain/docs/modules/
 'seq_num': 7, 'sender_name': 'User 2', 'timesta
      Document(page_content='Online is at least 
 '/Users/avsolatorio/WBG/langchain/docs/modules/
 'seq_num': 8, 'sender_name': 'User 2', 'timesta
      Document(page_content='How much do you wan
 '/Users/avsolatorio/WBG/langchain/docs/modules/
 'seq_num': 9, 'sender_name': 'User 1', 'timesta
      Document(page_content='Goodmorning! $50 is
 '/Users/avsolatorio/WBG/langchain/docs/modules/
 'seq_num': 10, 'sender_name': 'User 2', 'timest
      Document(page_content='Hi! Im interested i
 Thanks!', metadata={'source':
 '/Users/avsolatorio/WBG/langchain/docs/modules/
 'seq_num': 11, 'sender_name': 'User 1', 'timest
```

Now, you will see that the documents contain the metadata associated with the content we extracted.

The `metadata_func`

As shown above, the `metadata_func` accepts the default metadata generated by the `JSONLoader`. This allows full control to the user with respect to how the metadata is formatted.

For example, the default metadata contains the `source` and the `seq_num` keys. However, it is possible that the JSON data contain these keys as well. The user can then exploit the `metadata_func` to rename the default keys and use the ones from the JSON data.

The example below shows how we can modify the `source` to only contain information of the file source relative to the `langchain` directory.

```
# Define the metadata extraction function.
def metadata_func(record: dict, metadata:
dict) -> dict:

    metadata["sender_name"] =
record.get("sender_name")
    metadata["timestamp_ms"] =
record.get("timestamp_ms")

    if "source" in metadata:
        source =
metadata["source"].split("/")
        source =
source[source.index("langchain"):]
        metadata["source"] = "/".join(source)

    return metadata


loader = JSONLoader(

file_path='./example_data/facebook_chat.json',
    jq_schema='.messages[]',
    content_key="content",
    metadata_func=metadata_func
)

data = loader.load()
```

```
pprint(data)
```

```
    [Document(page_content='Bye!', metadata={'s
'langchain/docs/modules/indexes/document_loader
'seq_num': 1, 'sender_name': 'User 2', 'timesta
     Document(page_content='Oh no worries! Bye'
'langchain/docs/modules/indexes/document_loader
'seq_num': 2, 'sender_name': 'User 1', 'timesta
     Document(page_content='No Im sorry it was 
metadata={'source':
'langchain/docs/modules/indexes/document_loader
'seq_num': 3, 'sender_name': 'User 2', 'timesta
```

```
     Document(page_content='I thought you were 
'langchain/docs/modules/indexes/document_loader
'seq_num': 4, 'sender_name': 'User 1', 'timesta
     Document(page_content='Im not interested i
metadata={'source':
'langchain/docs/modules/indexes/document_loader
'seq_num': 5, 'sender_name': 'User 1', 'timesta
     Document(page_content='Here is $129', meta
'langchain/docs/modules/indexes/document_loader
'seq_num': 6, 'sender_name': 'User 2', 'timesta
     Document(page_content='', metadata={'sourc
'langchain/docs/modules/indexes/document_loader
'seq_num': 7, 'sender_name': 'User 2', 'timesta
     Document(page_content='Online is at least 
'langchain/docs/modules/indexes/document_loader
'seq_num': 8, 'sender_name': 'User 2', 'timesta
     Document(page_content='How much do you wan
'langchain/docs/modules/indexes/document_loader
'seq_num': 9, 'sender_name': 'User 1', 'timesta
     Document(page_content='Goodmorning! $50 is
'langchain/docs/modules/indexes/document_loader
'seq_num': 10, 'sender_name': 'User 2', 'timest
     Document(page_content='Hi! Im interested i
you are interested. Thanks!', metadata={'source
'langchain/docs/modules/indexes/document_loader
'seq_num': 11, 'sender_name': 'User 1', 'timest
```

Common JSON structures with jq

schema

The list below provides a reference to the possible `jq_schema` the user can use to extract content from the JSON data depending on the structure.

```
JSON        -> [{"text": ...}, {"text": ...}, {"text": ...}]
jq_schema   -> ".[].text"

JSON        -> {"key": [{"text": ...}, {"text": ...}, {"text": ...}]}
jq_schema   -> ".key[].text"

JSON        -> ["...", "...", "..."]
jq_schema   -> ".[]"
```