The LuaHTTP package

Johannes Casaburi
`johannes.casaburi@protonmail.com`

June 12, 2023
Version 1.0.1

Contents

1 Introduction 2

2 Manual Installation 2

3 Usage 3
- 3.1 fetchJson 3
- 3.2 fetchJsonUsingFile 4
- 3.3 fetchJsonUsingQuery 4
- 3.4 fetchRss 5
- 3.5 fetchImage 5

4 Examples 6
- 4.1 fetchJson 6
- 4.2 fetchJsonUsingFile 6
- 4.3 fetchJsonUsingQuery 7
- 4.4 fetchRss 7
- 4.5 fetchImage 7

1 Introduction

This is a small package that is the result of a case study on Internet-Interactive LaTeX PDF-documents. It provides five commands to make API calls, fetch data from the internet and insert that data into the PDF-document. This package depends on LuaTex, Lua and additional dependencies that need to be installed.

2 Manual Installation

First the required dependencies need to be installed:

`MacOS using Homebrew`

```
brew install lua@5.3 luarocks expat openssl
```

`Arch`

```
pacman -S lua53 luarocks expat openssl
```

`Debian`

```
apt install liblua5.3-dev libssl-dev lua5.3
   libexpat1-dev luarocks
```

Copy the `.sty` and `.lua` files in the directory where the `.tex` file is located.

In order to get the package working, a few dependencies first need to be installed. An easy way to do so, is by using `luarocks` which is the package manager for Lua modules which are called `rocks`.

To install the Lua rocks locally in the directory where the `.tex` file(s) is located, execute the following commands. This will initialize a new Lua project and install the required dependencies (See below for MacOS):

```
luarocks --lua-version=5.3 init
luarocks --lua-version=5.3 install dkjson
luarocks --lua-version=5.3 install luasec
luarocks --lua-version=5.3 install ltn12
luarocks --lua-version=5.3 install luaexpat
luarocks --lua-version=5.3 install feedparser
```

The Lua version may be changed if the Lua rocks support the version.

For `MacOS` you may need to add the Lua5.3 executable to your `$PATH`.

```
echo 'export␣PATH="/usr/local/opt/lua@5.3/bin:$PATH"' >> ~/.zshrc
```

And it may also be necessary to install `luaexpat` and `luasec` using the commands below:

```
luarocks install luaexpat EXPAT_DIR=/usr/local/Cellar/expat/${YOUR_VERSION_HERE}
luarocks install luasec OPENSSL_DIR=/usr/local/Cellar/openssl@3/${YOUR_VERSION_HERE}
```

3 Usage

The LaTeX package can be used by adding the following line to the `.tex` file.

```
\usepackage{luahttp}
```

To compile the PDF-document use:

```
lualatex --shell-escape
```

3.1 fetchJson

```
\fetchJson{URL}[optional: "key1,key2,.."]
```

This command takes an URL as the first argument and a list of comma separated keys as a second optional argument. Upon execution a GET request will be sent to the specified address. The accept header is set to `application/json`. The keys which are to be specified like `key1,key2,..` are used to filter out the desired values from the response. They have to exactly math the keys in the response otherwise the value will not be written to the PDF-Document. The same key can occur multiple times, this will output all values found to the PDF-Document.

Values with special characters are escaped. Values are searched for URLs. If an URL pointing to a image is detected the user will be prompted to either display the image or the actual URL.

3.2 fetchJsonUsingFile

```
\fetchJsonUsingFile{Path to JSON file}[optional: "key1,key2,.."]
```

This is very similar to the `fetchJson` command but takes a path to a JSON file instead of an URL. This allows the user to specify the request method, headers and body. The file should look something like this:

```
{
    "method": "POST",
    "url": "https://httpbin.org/post",
    "redirect": false,
    "headers": {
        "Accept": "application/json",
        "Content-Type": "application/json"
    },
    "body": {
        "name": "john",
        "id" : 1234
    }
}
```

Like the command before this command also takes and optional argument where the `keys` can be specified. If a value contains an URL leading to an image the user will be asked to either display the image or the URL.

3.3 fetchJsonUsingQuery

```
\fetchJsonUsingQuery{URL}{"key1,key2,.."} [optional: "?queryparameter1=value1"] .. [optional: "&queryparameter5=value5"]
```

This command can be used to send up to five query parameters using a POST request. The required arguments are the `URL` and a list of keys. The optional query parameters need to be written in key-value pairs:

```
\fetchJsonUsingQuery{URL}{key1,key2}[?q=The first
   value][&second=The second value and so on"]
```

The values after the `=` sign are URL-encoded and sent using the `application/json` Accept header and `application/x-www-form-urlencoded` Content-Type header. Like in the `fetchJson` command the user will be prompted if an URL leading to an image is detected.

3.4 fetchRss

```
\fetchRss{URL}{limit}[optional: "feedinfokey1,
   feedinfokey2,.."][optional: "entrykey1,entrykey
   2,.."]
```

This command takes an `URL` and a `limit`. The limit describes how many entries of the feed are written to the PDF-Document. There are two optional arguments. The first argument is a list of keys used to filter out feed information. At this time possible values are `title, subtitle, rights, generator, author, author_detail, link, links, updated_parsed, updated, id, contributors`. Every entry can be filtered using the second argument. Possible values for the entry keys are `id, link, links, updated, updated_parsed, title, summary, contributors`. These values may change if the Lua rock called `feedparser` that is used to parse the feed changes. More information about `feedparser` can be found here: `https://github.com/LuaDist-testing/feedparser`

The `summary` from the entries may contain HTML which will just be written to the PDF-Document. As described in the `fetchJson` command you may be promoted when a link to an image is detected.

3.5 fetchImage

```
\fetchImage{URL}[optional: width in cm][optional:
   height in cm]
```

Images can be inserted into the PDF-Document using the URL. To resize the image, one or two optional arguments can be given to specify the width or width and height in `centimeters`:

```
\fetchImage{URL}[7cm][5cm]
```

4 Examples

4.1 fetchJson

To display the title and summary of the first article returned by the spacenews API we have to specify the URL and the keys that hold the desired values:

```
\fetchJson{https://api.spaceflightnewsapi.net/v3/articles?_limit=1}[title,summary]
```

4.2 fetchJsonUsingFile

Like the example above the same can be done using `fetchJsonUsingFile`. For that we need a `JSON` file with the following content:

```
{
    "method": "GET",
    "url": "https://api.spaceflightnewsapi.net/v3/articles?_limit=1",
    "redirect": false,
    "headers": {
        "Accept": "application/json"
    }
}
```

If the `JSON` file is located in the same directory as the `.lua` files the command looks like this:

```
\fetchJsonUsingFile{myjsonfile.json}[title,summary]
```

If the package was not manually installed the absolute path must be used.

4.3 fetchJsonUsingQuery

Here is an example of an API call using query parameters:

```
\fetchJsonUsingQuery{http://127.0.0.1:5000/translate
   }{translatedText}[?q=I want to translate this
   text][&source=en][&target=de][&format=text]
```

First we need to specify the first part of the URL, followed by the key(s) that hold the desired return values. The values of the query parameters get URL-encoded and put back together in a single URL. We may need to add an `&` before the query parameter or an `?` depending on the location.

4.4 fetchRss

In this example the first the subtitle from the feed information is printed as a subsection. The second command prints the `title,` `link` and `updated` information of the first three entries received.

```
\subsection{
    \fetchRss{https://www.eff.org/rss/updates.xml
   }{1}[subtitle][none]
}

\fetchRss{https://www.eff.org/rss/updates.xml}{3}[
   none][title,link,updated]
```

Note that `none` is not a special keyword, if there was a key called `none` then its value would be printed to the PDF-Document.

4.5 fetchImage

Here is an example how to embed an image and scaling it to have a width of 5cm:

```
\fetchImage{https://i0.wp.com/spacenews.com/wp-
   content/uploads/2023/03/vorb-march2023.jpg}[5cm]
```