

Markdown Converter

Author

Mehmet Fatih SARAÇ

A terminal-based Markdown (.md) to HTML (.html) / text file (.txt) / string (str) converter which handles paragraphs, headings (h1 to h6), **bold**, *italic* and ~~strikethrough~~ texts, [links](#), images, unordered lists^[1] and codes. The project is based on GitHub Markdown, but it can convert any *relatively simple* Markdown to HTML.

Introduction

Aim

The aim of this project is to convert the Markdown markup language, which is frequently used especially when writing the documentation of a tool, into raw HTML and then output it as different file types (html, txt or str) according to the user's request.

Scope

In this project, the user gives a Markdown file to the program. The program first converts this file into raw HTML. After this process, depending on the output type requested by the user, the program prints this raw HTML as that file.

Methodology

In this project I mostly utilized "functional programming", which is a declarative programming paradigm style. I used regular expressions (also known as regex), a very powerful tool to convert Markdown lines into HTML blocks. I also used Python's built-in "argparse" library to give the user a little more flexibility in the terminal.

^[1]: Converting unordered lists has some issues. Converter adds tags but does not add tag at the start and end of list items. Thus, the list is displayed without indentation from right side.

Theoretical Part

To convert the Markdown file into raw HTML, we first need to understand the Markdown syntax and then know the HTML equivalent of each Markdown mark.

- Headers: # for <h1>, ## for <h2>, etc.
- Italic: **italic** or _italic_ for
- Bold: ****bold**** or __bold__ for .
- Strikethrough: ~~~~strikethrough~~~~ for <s>
- Links: [link](http://url) for link
- Images: ![alt text](url) for
- Unordered lists: -, * or + for
- Code: ``code`` for <code>

After this conversion, we need to print the raw HTML we obtain as a file type according to the user's request. For this, we need to be able to open and write files.

Practical Part

The main part of the project is shown below:

```
def convert_to_raw_html(content):  
    # Convert to paragraph  
    content = re.sub(r"^([^#!*_\-\+\n].*)$", r"<p>\1</p>", content,  
flags=re.MULTILINE)  
  
    # Convert to headings  
    content = re.sub(  
        r"^(#{1,6}) +(.*)$",  
        lambda m: f"<h{len(m.group(1))}>{m.group(2)}</h{len(m.group(1))}>",  
        content,  
        flags=re.MULTILINE  
    )  
  
    # Convert to bold  
    content = re.sub(  
        r"\*(?2)([^\*]+)\*(?2)", r"<strong>\1</strong>", content,  
        flags=re.MULTILINE  
    )  
  
    # Convert to italic
```

```

    content = re.sub(r"_(^_+)_(:? |\n)", r"<em>\1</em> ", content,
flags=re.MULTILINE)

    # Convert to strikethrough
    content = re.sub(r"~{2}([^~]+)~{2}", r"<s>\1</s>", content,
flags=re.MULTILINE)

    # Convert to link
    content = re.sub(
        r"\[([^\]]+)\]\(\(.+?(?=\\ )\)\)",
        r'<a href="\2">\1</a>',
        content,
        flags=re.MULTILINE,
    )

    # Convert to image
    content = re.sub(
        r"!\[([^\]]*)\]\(\(.+?(?=\\)\n)\)",
        r'',
        content,
        flags=re.MULTILINE,
    )

    # Convert to unordered lists
    content = re.sub(r"^(?:-|\*|\+)(.*)$", r"<li>\1</li>", content,
flags=re.MULTILINE)

    # Convert to code
    content = re.sub(r"`([^\`]+)`", r"<code>\1</code>", content,
flags=re.MULTILINE)

    return content

```

The above code basically does the job of converting the Markdown string into raw HTML string, which I explained in the theoretical part.

One of the shortcomings of the project that is difficult to understand is that it does not add tags when converting the unordered list. However, I couldn't find how to do this easily.

Another shortcoming of the project is that it is not a full-fledged Markdown converter.

Summary

In summary coding a Markdown converter using regular expressions (regex) was a little more difficult than I expected. This version of the project has a few “minor” bugs and many shortcomings. I implemented the most used functions in Markdown in this project, but if you want to use a real Markdown converter in a real project I recommend using the [python-markdown2](#) library instead of mine.