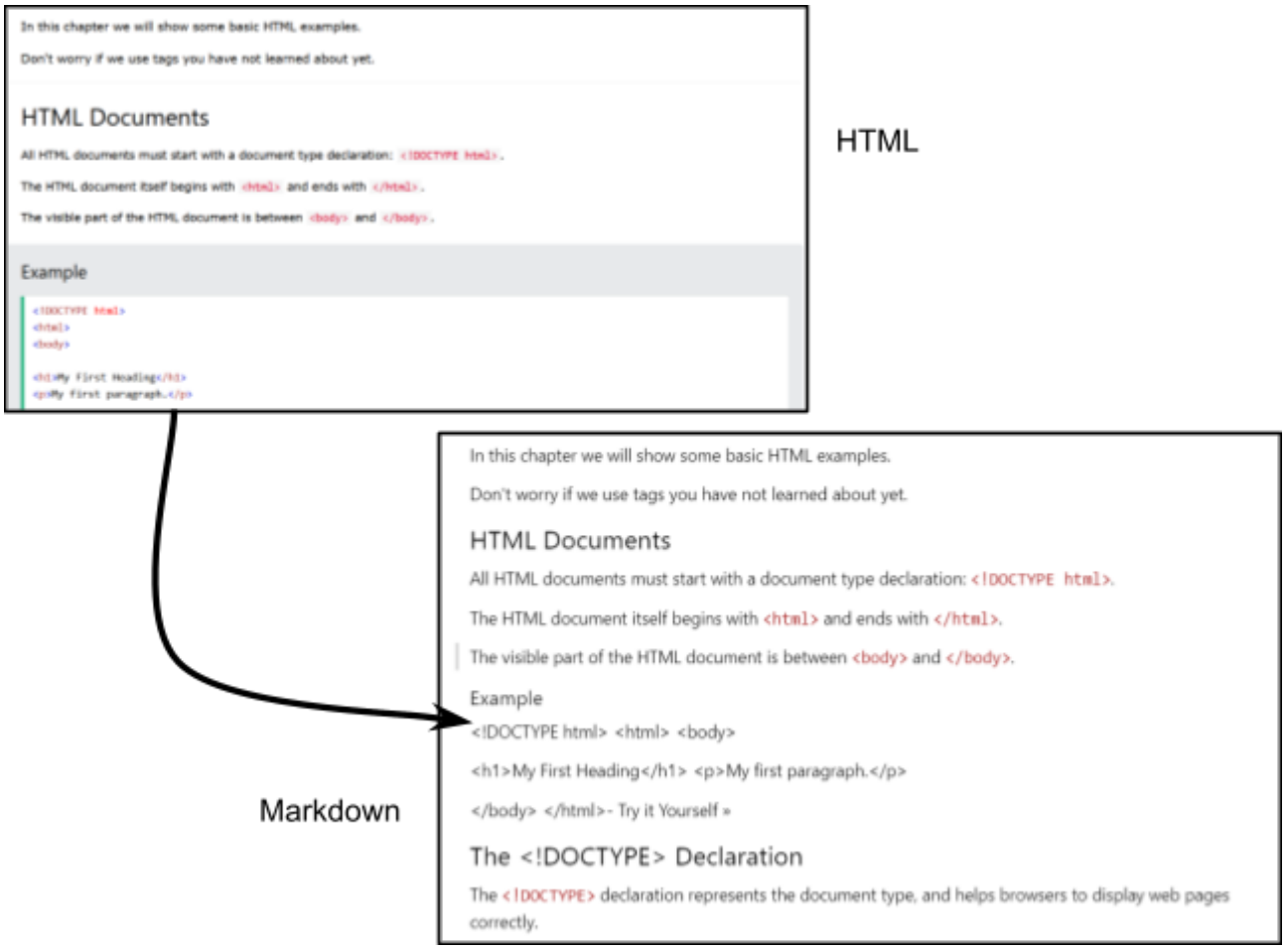
Generador de Documentación en Markdown desde Páginas Web mediante Flex

Shao Jie Hu Chen, César Muñoz Reinoso
Doble Grado en Ingeniería Informática y Matemáticas
Curso 2022/2023. Modelos de Computación.



1. Introducción

La consulta de documentación técnica en diferentes webs es de especial relevancia en el ámbito de la informática. Ya sea para buscar una información específica de una librería o, especialmente, el aprendizaje de un nuevo lenguaje de programación, es usual acudir a la información presente en páginas web como GeeksForGeeks, W3School, Codecademy, etc.

Si queremos realizar anotaciones sobre el contenido presente en estas páginas tenemos ciertos inconvenientes. Por un lado, no permite realizar anotaciones directamente sobre el contenido expuesto, lo que impide poder consultar las dudas que nos hubiere generado previamente. Por otra parte, la consulta requiere del uso de un navegador web, lo cual puede no ser rápido para realizar estas acciones, puesto que o bien tenemos almacenado previamente el enlace a la página web o bien se ha de encontrar la página mediante un buscador.

Para solventar estos problemas, hemos empleado la herramienta Flex para, pasando un archivo HTML como los proporcionados por páginas web como W3School, podamos generar una documentación en Markdown, mucho más cómodo para estas labores y permite realizar fácilmente anotaciones sobre el contenido expuesto en estas páginas.

2. Implementación

Si bien es cierto que HTML es un lenguaje independiente del contexto (pero no regular), una escritura del código HTML suficientemente estructurado permite extraer expresiones regulares útiles para nuestro propósito. Para conseguir esta estructuración, se puede emplear herramientas como FreeFormatter, que permite transformar un código HTML en una estructura más natural.

Con un fichero HTML bien estructurado, nuestro código detecta los siguientes patrones:

- **Encabezados**. Es el texto comprendido entre <h1> y </h1>; <h2> y </h2>; …; <h6> y </h6> (sólo se establecen 6 niveles de encabezado en HTML). Nuevamente, definimos una regla de forma que capture la información comprendida entre los tokens y lo imprima en un fichero, precedido de los caracteres ##, ###, etc. (el que proceda). En la implementación, se ha realizado cada uno de forma independiente, puesto que si hubiéramos puesto una expresión regular de la forma <h{DIGITO}> y </h{DIGITO}>, no podríamos asegurar que en ambos casos sea el mismo dígito, por lo que el código intermedio se habría procesado erróneamente.

- **Párrafos**. Es el mismo caso que los anteriores, pero en este caso es el texto comprendido entre y . Se imprime directamente en el fichero, procurando dejar una línea en blanco tanto antes como después.

- **Tablas**. Es el mismo caso que los anteriores, pero con <table> y </table>. En este caso, también copiamos el contenido en el fichero (Markdown detecta la sintaxis de la tabla en HTML).

- **Código**. Es el texto comprendido entre <code> y </code>. En este caso, como Markdown detecta adecuadamente esta misma sintaxis, se reproduce tal cual en el fichero de salida.

- **Otra información**. Las otras informaciones no proporcionan datos útiles para la documentación, por lo que las descartamos.

Se suministra el código del generador en el fichero `HtmlToMd.l`. En ella, reemplazamos los patrones dichos y lo transformamos en su equivalente en Markdown.

3. Uso

3.1. Compilación

Para compilar el fichero `HtmlToMd.l` en Ubuntu 22.04, se ha de tener instalado la utilidad `flex++`. Para generar un archivo en C++ a partir de nuestro fichero HtmlToMd.l, se emplea el siguiente comando:

```
flex++ -o HtmlToMd.yy.cpp HtmlToMd.l
```

El comando anterior crea un fichero en C++ llamado `HtmlToMd.yy.cpp`, a partir del cual podemos generar un ejecutable.

```
g++ -std=c++11 -o HtmlToMd HtmlToMd.yy.cpp
```

En este caso, hemos llamado a nuestro ejecutable `HtmlToMd`.

3.2. Ejecución

Para ejecutar nuestro programa, habiendo seguido las instrucciones de compilación anteriores, hemos de seguir la siguiente estructura:

```
./HtmlToMd <fichero_html> <salida_md>
```

Como ejemplo, se suministra un fichero `fichero_simple.html` para probar el funcionamiento del comando:

```
./HtmlToMd fichero_simple.html salida_simple.md
```

Esto genera un archivo llamado `salida_simple.md` que produce el resultado de procesar el archivo HTML. Su contenido se muestra a continuación:

```
...
# My First Heading


## Form Events

My first __palabra__ paragraph.

...
## Form Events

My fourth __palabra__ paragraph.
```

```
...

##### Form Events


###### Form Events
```

4. Ejemplo de uso práctico: documentación en W3School

Este programa nos será útil para obtener la información de una documentación en W3School para pasarlo directamente a un fichero Markdown en nuestra máquina. Nos descargamos un fichero correspondiente a la documentación de HTML sobre ejemplos básicos de uso en W3School. Después de pasarlo por la herramienta de formateo, proporciona una página que no es del todo legible, por lo que se ha modificado ligeramente para que saliese una página limpia (se suministra por PRADO). Posteriormente, ejecutamos el siguiente comando:

```
./HtmlToMd fichero_w3school.html salida_w3school.md
```

El fichero de salida también se proporciona en la entrega del trabajo. En ella, podemos ver cómo se transforma un fichero desde una página web hasta el fichero Markdown.

Un ejemplo de entrada del fichero HTML suministrado es el siguiente fragmento:

```
...

<h2>HTML Documents</h2>
<p>All HTML documents must start with a document type declaration: <code
class="w3-codespan">&lt;!DOCTYPE html&gt;</code>.</p>

<p>The HTML document itself begins with <code class="w3-codespan">&lt;html&gt;</code> and
ends with <code class="w3-codespan">&lt;/html&gt;</code>.</p>

<p>The visible part of the HTML document is between <code
class="w3-codespan">&lt;body&gt;</code> and <code
class="w3-codespan">&lt;/body&gt;</code>. </p>
...
```

El texto anterior se transforma en el siguiente código (también proporcionado en el fichero zip entregado):

```
...

## HTML Documents


All HTML documents must start with a document type declaration: <code
class="w3-codespan">&lt;!DOCTYPE html&gt;</code>.

The HTML document itself begins with <code class="w3-codespan">&lt;html&gt;</code> and
ends with <code class="w3-codespan">&lt;/html&gt;</code>.

The visible part of the HTML document is between <code
class="w3-codespan">&lt;body&gt;</code> and <code
class="w3-codespan">&lt;/body&gt;</code>.
```

. . .

De esta manera, y con todo el código presente en HTML, se produce un fichero de salida en formato Markdown con la documentación presente en la página web, como se deseaba. De esta manera, podemos tomar anotaciones más cómodamente sobre la documentación en W3School.