HTML to LaTeX transformation

Frederik R. N. Schlupkothen

Abstract

(LA)TEX was created as an authoring language that enables authors to keep typesetting quality standards while preparing their printed matter, assuming that the output context is known from the very beginning of the writing process. As a counter-concept, XML is focused on keeping output flexible, providing mechanisms to manage and control the logical structure of documents. Combining the strengths of both ecosystems has been discussed frequently in the past. This article aims to contribute to this discussion by introducing a mapping from HTML to LATEX, the two most widespread document description languages in their respective fields of application.

1 Introduction

The Extensible Markup Language (XML) has become the native markup language for structured information in (distributed) document processing architectures. It provides a core syntax that has been adopted by many document description languages. A broad software ecosystem and further standards have emerged to realize and facilitate the processing of XML-based documents. Among them, the Extensible Stylesheet Language Transformations (XSLT) described in [8] offers a standardized mechanism to translate different XML-based languages into one another. This has made XML the language of choice for cross-media publishing workflows.

However, while the XML processing is fully covered by viable tools, the final document production of printed matter from XML may still be a problem. A potential solution is to integrate the TEX typesetting engine (described in [10]) in XML-based processing chains. For this task, TEXML, an XML representation of TEX commands, was introduced in [12] and its "production proof" implementation discussed in [14]. As any other XML language, TEXML documents can be produced via XSLT. So the last gap to fill in TEXML-based workflows is to define XSL stylesheets that realize the transformation between specific XML and TEX document description languages (see [14]).

Here we introduce a mapping between two document description languages that are well known in their respective fields of application: The HyperText Markup Language (HTML), as the transformation's source language, is the core language of the World Wide Web and has a history that is closely tied to XML. It offers layout-oriented markup semantics primarily for textual content and is used amongst others

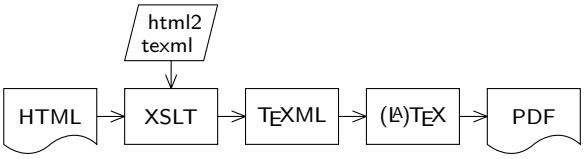


Figure 1: HTML to PDF processing

to describe web pages (see [7]), electronic books (see [4]), and printed matter (see e.g. [9, 13]). LATEX, as the transformation's output, is the abstraction of TEX's typesetting commands to logical markup.

Figure 1 shows the XML-based production workflow that produces the Portable Document Format (PDF) from HTML by the use of the TEX typesetting engine. The particular processing steps are described via an example of emphasized text:

1. The HTML source document describes the emphasized text "dolor" as: `<em>dolor</em>`
2. The XSLT processor queries the stylesheet "html 2 texml" for a matching transformation template. The matching template defines the transformation of the HTML's "em" element to the corresponding LATEX command in its TEXML representation:
```
<xsl:template match="html:em">
  <tex:cmd name="emph">
    <tex:parm>
      <xsl:apply-templates />
    </tex:parm>
  </tex:cmd>
</xsl:template>
```
3. The result of the XSLT transformation is the following TEXML element:
```
<cmd name="emph">
  <parm>dolor</parm>
</cmd>
```
4. The TEXML processor converts the TEXML element to a LATEX command: `\emph{dolor}`
5. The LATEX processor typesets: *dolor*

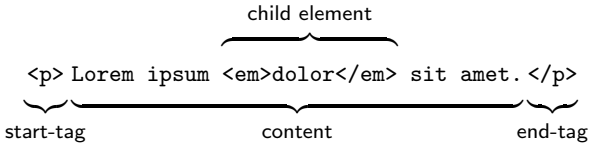
The prior example shows that defining the transformation between two languages needs insight into the differences in conceptual syntax and semantic coverage of source and destination language. While HTML is native to the XML syntax, TEXML is reproducing the syntax logic inherited from (LA)TEX. While LATEX is native to printed matter, HTML was initially designed for electronic resources. Section 2 introduces the underlying concepts of the HTML and LATEX markup syntax. Section 3 introduces the mapping between HTML and LATEX markup semantics. Finally, section 4 shows a complete document with its corresponding representations in HTML and LATEX.

2 Markup syntax

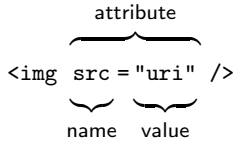
The following two subsections give an overview of the very basic HTML and LaTeX syntax. They do not introduce the full syntax but focus on the aspects needed within this article. Full descriptions can be found in [3] for XML, the underlying markup language of HTML as described in [6], and in [10, 11] for LaTeX.

2.1 Basic syntax of HTML

An HTML document consists of *elements* that are either *empty* or *non-empty*. The boundaries of a non-empty element is marked by a *start-tag* and a *end-tag*. Tag delimiters are the `<` and `>` characters. The element *type* is defined in the start-tag by its *name*. The end-tag repeats the element name preceded by a `/` character. The element's *content* is enclosed between the start- and end-tag. The content consists of *character data* (i.e. text), subordinated *child* elements, or both. An element without content is called empty and is either described by a start-tag that is directly followed by its end-tag or by an *empty-element-tag*. An empty-element-tag has the same form as a start-tag but ends with a `/` character. The following example shows a non-empty 'p'-element with mixed content:



An element can possess *attributes*. Attributes are noted in the start-tag or empty-element-tag behind the element name. Attributes consist of a *name-value*-pair. The attribute-value is given between two `'` or `"` characters and is assigned to its name by a preceding `=` character. The following example shows an empty 'img' element with a 'src' attribute of the value 'uri':

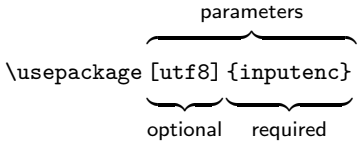


There is exactly one *root* element that includes all the document's content. The tag placement within the document follows the rules of mathematical brackets. The examples below show possible tag placements by means of 'a' and 'b' elements:

sequence	`<a></a><b></b>`
subordination	`<a><b></b></a>`
syntax error	`<a><b></a></b>`

2.2 Basic syntax of LaTeX

A LaTeX document consists of *commands* that describe either output characters (i.e. characters to typeset), special characters (e.g. the `~` character for a non-breaking space), or *control sequences*. There are two types of control sequences: *control words* and *control symbols*. A control word starts with a `\` character followed by its *name* that consists of one or more letters (i.e. lower- or uppercase letters 'a' to 'z') and is terminated by either a space or another non-letter. A control symbol starts with a `\` character followed by one non-letter. A command can possess optional and required *parameters* that are set by *arguments*. Optional parameter arguments are noted after the command name between square brackets, and required parameter arguments between curly braces. The following example shows a 'usepackage'-command with an optional parameter set to 'utf8' and a required parameter set to 'inputenc':



Furthermore there are two special types of commands: *environments* and *declarations*. Environments are pairs of 'begin'- and 'end'-commands that enclose the environment's content. The environment name is provided as the first required argument of the corresponding 'begin'- and 'end'-commands. The arguments of the environment are noted as further arguments of the 'begin'-command. Declarations influence the behavior of following commands. The *scope* (i.e. range of effect) of most declarations is limited to its enclosing environment or *group*. The group delimiters are the `{` and `}` characters. The placement of group delimiters and environment commands follows the rules of mathematical brackets. The examples below show possible placements by example of a group and an 'x'-environment:

sequence	`{ } \begin{x}\end{x}`
subordination	`{ \begin{x}\end{x} }`
syntax error	`{ \begin{x} }\end{x}`

3 Markup correspondence

The following sections introduce a possible mapping between HTML elements and LaTeX commands in the order of the HTML module descriptions in [1]. For an XSLT implementation transforming HTML to TeXML, the following mapping tables show the resulting LaTeX commands for expository purposes.

3.1 Core Modules

The HTML Core Modules assemble the markup that is common to all HTML dialects that are derived from module-based HTML. This core markup for high level structures, basic text, hyperlinks, and lists of HTML documents and its corresponding LaTeX commands are described in the following subsections.

3.1.1 Structure Module

The HTML Structure Module defines the high level markup of a document. The `html`-element is the document's root containing the meta-information (`head`) and the actual content (`body`) of a document. LaTeX follows a similar separation with its preamble and `document`-environment. Table 1 below shows the corresponding commands.

Table 1: HTML to LaTeX structure mapping

HTML	LaTeX
`<html>`⟨...⟩	`\documentclass{report}`⟨...⟩
`<head>`⟨...⟩	⟨...⟩
`<title>`⟨...⟩	`\title{`⟨...⟩`}`
`<body>`⟨...⟩	`\begin{document}`⟨...⟩

3.1.2 Text Module

The HTML Text Module defines the basic text markup to describe heading, block, and inline elements. Most of these elements have equivalent commands in LaTeX, but not all. In these cases the '↦' symbol indicates the default formatting in HTML where the Presentation Module described in section 3.2.1 might be used for an alternative, not corresponding semantically, mapping.

Headings The HTML Text Module defines six levels of headings (`h1` to `h6`). LaTeX offers a specific heading hierarchy that depends on the given document class. Table 2 below shows the corresponding heading commands for the report document class.

Table 2: HTML to LaTeX heading mapping

HTML	LaTeX
`<h1>`⟨...⟩	`\chapter{`⟨...⟩`}`
`<h2>`⟨...⟩	`\section{`⟨...⟩`}`
`<h3>`⟨...⟩	`\subsection{`⟨...⟩`}`
`<h4>`⟨...⟩	`\subsubsection{`⟨...⟩`}`
`<h5>`⟨...⟩	`\paragraph{`⟨...⟩`}`
`<h6>`⟨...⟩	`\subparagraph{`⟨...⟩`}`

Blocks The HTML Text Module defines elements to mark text groups as paragraphs (`p`), contact information (`address`), quotations (`blockquote`), generic groups (`div`), and preformatted text (`pre`). Table 3 shows the corresponding LaTeX commands (using the LaTeX core package alltt for preformatted text).

Table 3: HTML to LaTeX block mapping

HTML	LaTeX
`<p>`⟨...⟩	⟨...⟩ `\par`
`<address>`⟨...⟩	↦ *italic*
`<blockquote>`⟨...⟩	`\begin{quote}`⟨...⟩
`<div>`⟨...⟩	⟨...⟩
`<pre>`⟨...⟩	`\begin{alltt}`⟨...⟩

Inlines The HTML Text Module defines markup for text fragments. This includes abbreviations (`abbr`) and acronyms (`acronym`), citations (`cite`), quotations (`q`), and definitions (`dfn`), program code (`code`), sample output (`samp`), arguments (`var`), and input (`kbd`), regular (`em`) and strong (`strong`) emphases, generic fragments (`span`), and forced line breaks (`br`). Table 4 below shows the corresponding LaTeX commands (using the glossaries package for abbreviations and acronyms, the csquotes package for quotations, and the listings package for code).

Table 4: HTML to LaTeX inline mapping

HTML	LaTeX
`<abbr>`⟨...⟩	`\acrshort{`⟨...⟩`}`
`<acronym>`⟨...⟩	`\ac{`⟨...⟩`}`
`<cite>`⟨...⟩	`\cite{`⟨*gen-id*⟩`}` `\bibitem{`⟨*gen-id*⟩`}`⟨...⟩
`<q>`⟨...⟩	`\enquote{`⟨...⟩`}`
`<dfn>`⟨...⟩	↦ *italic*
`<code>`⟨...⟩	`\lstinline`|⟨...⟩|
`<samp>`⟨...⟩	↦ *teletype*
`<var>`⟨...⟩	↦ *italic*
`<kbd>`⟨...⟩	↦ *teletype*
`<em>`⟨...⟩	`\emph{`⟨...⟩`}`
`<strong>`⟨...⟩	↦ *bold*
`<span>`⟨...⟩	⟨...⟩
` `	`\newline`

3.1.3 Hypertext Module

The HTML Hypertext Module defines markup to describe hyperlinks. They are described by source anchors (`a`) that reference to contents inside or outside of the document via Unified Resource Identifiers (URIs). Referenceable document fragments are marked by common 'id' attributes that can be applied to all elements. The use of traversable hyperlinks is an adequate solution in the context of electronic documents; its mapping to corresponding

LaTeX commands by means of the hyperref package is shown in Table 5. However, in the context of printed matter a solution with references by e.g. visual key or page numbers might be more appropriate.

Table 5: HTML to LaTeX hypertext mapping

HTML	LaTeX
`<a href="`⟨*uri*⟩`">`⟨...⟩	`\href{`⟨*uri*⟩`}{`⟨...⟩`}`
`<a href="`⟨*id*⟩`">`⟨...⟩	`\hyperref[`⟨*id*⟩`]{`⟨...⟩`}`
`id="`⟨*id*⟩`"`	`\label{`⟨*id*⟩`}`

3.1.4 List Module

The HTML List Module defines markup to describe ordered (`ol`) and unordered (`ul`) lists as sequences of list items (`li`) and furthermore markup to describe definition lists (`dl`) that are composed of sequences of term (`dt`) and description (`dd`) pairs. Table 6 below shows corresponding LaTeX commands.

Table 6: HTML to LaTeX list mapping

HTML	LaTeX
`<ol>`⟨...⟩	`\begin{enumerate}`⟨...⟩
`<ul>`⟨...⟩	`\begin{itemize}`⟨...⟩
`<li>`⟨...⟩	`\item` ⟨...⟩
`<dl>`⟨...⟩	`\begin{description}`⟨...⟩
`<dt>`⟨...⟩	`\item[`⟨...⟩`]`
`<dd>`⟨...⟩	⟨...⟩`\\`

3.2 Text Extension Modules

The HTML Text Extension Modules assemble additional text markup to control text rendering, maintenance, and direction for HTML documents. These and the corresponding LaTeX commands are described in the following subsections.

3.2.1 Presentation Module

The HTML Presentation Module defines markup to control the text rendering. It provides elements to render text in/as bold (`b`) and italic (`i`) style, typewriter (`tt`), super- (`sup`) or subscripted (`sub`), larger (`big`) or smaller font (`small`). Additionally the module provides an element to render horizontal rules (`hr`). LaTeX offers corresponding commands with the exception of 'textsubscript' that relies on the subscript package. The relsize package offers commands to realize relative font sizes (as intended by the 'big' and 'small' elements in HTML). Table 7 shows a possible mapping.

3.2.2 Edit Module

The HTML Edit Module defines editing-related markup. It provides elements to mark content as deleted (`del`) or inserted (`ins`). The changes package offers semantically corresponding LaTeX commands as shown in Table 8 below. However, if LaTeX is used as final output format only, a more stable solution might be to simply output contents of 'ins'-elements, but not those of 'del'-elements.

Table 7: HTML to LaTeX presentation mapping

HTML	LaTeX
`<b>`⟨...⟩	`\textbf{`⟨...⟩`}`
`<i>`⟨...⟩	`\textit{`⟨...⟩`}`
`<tt>`⟨...⟩	`\texttt{`⟨...⟩`}`
`<sup>`⟨...⟩	`\textsuperscript{`⟨...⟩`}`
`<sub>`⟨...⟩	`\textsubscript{`⟨...⟩`}`
`<big>`⟨...⟩	`{\larger` ⟨...⟩`}`
`<small>`⟨...⟩	`{\smaller` ⟨...⟩`}`
`<hr />`	`\hrulefill`

Table 8: HTML to LaTeX edit mapping

HTML	LaTeX
`<del>`⟨...⟩	`\deleted{`⟨...⟩`}`
`<ins>`⟨...⟩	`\added{`⟨...⟩`}`

3.2.3 Bi-directional Text Module

The HTML Bi-directional Text Module defines markup to declare text direction changes. It provides an attribute to control the direction of text (`dir`) that can be applied to all elements including a special element (`bdo`) to override the current text direction. The bidi package offers corresponding LaTeX commands. Table 9 below shows the corresponding commands for inline text. However, the bidi package defines a set of new environments which replace common LaTeX commands (e.g. lists and footnotes) which makes the general mapping between elements and commands more complex. Furthermore the combination with other common packages (e.g. hyperref or longtable) remains problematic. So a more stable solution might be to omit bi-directional text controls during the transformation process and to apply such changes manually in the LaTeX document.

Table 9: HTML to LaTeX bidi mapping

HTML	LaTeX
`<bdo dir="ltr">`⟨...⟩	`\LR{`⟨...⟩`}`
`<bdo dir="rtl">`⟨...⟩	`\RL{`⟨...⟩`}`

3.3 Forms Modules

The HTML Forms Modules define markup to describe interactive forms that can define, organize, and receive (textual) input and selections. The hyperref

package implements most HTML form elements for LaTeX. As with hyperlinks, the use of interactive forms is adequate for electronic documents; their mapping by means of the hyperref package is shown in Table 10 below. However, in the context of printed matter, an alternative solution as given e.g. by the formular package might be more suitable.

Table 10: HTML to LaTeX forms mapping

HTML	LaTeX
`<form>`⟨...⟩	`\begin{Form}`⟨...⟩
`<input />`	`\TextField{`⟨*label*⟩`}`
`type="password"`	`\TextField[password]{`⟨*label*⟩`}`
`type="checkbox"`	`\CheckBox{`⟨*label*⟩`}`
`type="button"`	`\PushButton{`⟨*label*⟩`}`
`type="radio"`	`\ChoiceMenu[radio]{`⟨*label*⟩`}{=}`
`type="submit"`	`\Submit{`⟨*label*⟩`}`
`type="reset"`	`\Reset{`⟨*label*⟩`}`
`type="file"`	`\TextField[fileselect]{`⟨*label*⟩`}`
`type="hidden"`	`\TextField[hidden]{`⟨*label*⟩`}`
`type="image"`	`\Submit[submitcoordinates]{`⟨*img*⟩`}`
`<select>`⟨...⟩	`\ChoiceMenu{`⟨*label*⟩`}{`⟨*options*⟩`}`
`<option>`⟨...⟩	⟨...⟩
`<textarea>`⟨...⟩	`\TextField[multiline]{`⟨*label*⟩`}`
`<button>`⟨...⟩	`\Submit{`⟨...⟩`}`
`type="button"`	`\PushButton{`⟨...⟩`}`
`type="reset"`	`\Reset{`⟨...⟩`}`
`<fieldset>`⟨...⟩	⟨...⟩
`<label>`⟨...⟩	⟨...⟩
`<legend>`⟨...⟩	⟨...⟩
`<optgroup>`⟨...⟩	⟨...⟩

3.4 Table Modules

The HTML Table Modules define markup to describe tables (`table`) by organizing their data (`td`) and header (`th`) cells in rows (`tr`). These rows can be grouped into table headers (`thead`), footers (`tfoot`), and bodies (`tbody`). Column-based markup is realized by standoff elements (`col` and `colgroup`). A table caption (`caption`) can provide a short description of the table contents.

LaTeX table definitions differ in two essential aspects from HTML: *(i)* The total number of table columns has to be given explicitly to a LaTeX table environment. This is not necessary in HTML but calculated continuously by the rendering engine at processing time. *(ii)* LaTeX table cells that span several rows (by means of the multirow package) cover the adjacent cells in the following rows; therefore empty cells need to be inserted in the following rows. This is not necessary in HTML but the rendering engine automatically shifts the cells of the following rows according to the reading direction.

$table = \{row_i \mid row_i = \{cell_{ij}\}\}$
$grid = \{slot_i \mid slot_i = \langle x, y\rangle\}$
procedure TABLE($table$)
  $grid \leftarrow \{\emptyset\}$
  for all $row_i \mid i = 1\,..\,n$ **do**
    $y \leftarrow i$
    $x \leftarrow 1$
    for all $cell_{ij} \mid j = 1\,..\,m$ **do**
      while $grid \ni \langle x, y\rangle$ **do**
        EMPTY CELL(x, y)
        $x \leftarrow x + 1$
      end while
      for $y_{\text{cell}} \leftarrow 0\,..\,\text{rowspan}(cell_{ij}) - 1$ **do**
        for $x_{\text{cell}} \leftarrow 0\,..\,\text{colspan}(cell_{ij}) - 1$ **do**
          $grid \leftarrow grid \cup \langle\, x + x_{\text{cell}}\,,\, y + y_{\text{cell}}\,\rangle$
        end for
      end for
      $x \leftarrow x + \text{colspan}(cell_{ij})$
    end for
  end for
end procedure

Figure 2: HTML table cell positioning algorithm

Hence for the transformation of HTML tables to LaTeX this information (total number of table columns and position of additional empty cells) need to be precalculated. Therefore the transformation process has to include parts of the HTML table processing model described in [2]. This model describes an HTML table as a set of cells that are positioned on a two-dimensional grid of slots. The algorithm shown in Figure 2 calculates the cell positioning and illustrates how the additional empty cells are inserted; hence the total number of table columns is given by the maximum x-coordinate within the final grid. Table 11 shows the mapping of HTML table elements to corresponding LaTeX commands by means of the longtable package.

3.5 Image Module

The HTML Image Module defines markup to embed external images. The graphicx package offers a corresponding LaTeX command as shown in Table 12.

3.6 Further Modules

The HTML specification describes further modules that define markup to realize dynamic and interactive document content, mechanisms to control layout, and deprecated markup for backwards compatibility with legacy HTML. Due to the focus of this article on the transfer of the logical structure of HTML documents

Table 11: HTML to LaTeX table mapping

HTML	LaTeX
`<caption>`⟨...⟩	`\caption{`⟨...⟩`}`
`<table>`⟨...⟩	`\begin{longtable}{`⟨*col*⟩`}`⟨...⟩
`<td>`⟨...⟩	⟨...⟩`&`
`<th>`⟨...⟩	`\bf{}`⟨...⟩`&`
`colspan="`⟨*span*⟩`"`	`\multicolumn{`⟨*span*⟩`}{l}{`⟨...⟩`}`
`rowspan="`⟨*span*⟩`"`	`\multirow{`⟨*span*⟩`}{*}{`⟨...⟩`}`
`<tr>`⟨...⟩	⟨...⟩`\\`
`<col />`	
`<colgroup>`⟨...⟩	
`<tbody>`⟨...⟩	⟨...⟩
`<thead>`⟨...⟩	⟨...⟩`\endhead`
`<tfoot>`⟨...⟩	⟨...⟩`\endfoot`

Table 12: HTML to LaTeX image mapping

HTML	LaTeX
`<img src="`⟨*uri*⟩`"/>`	`\includegraphics{`⟨*uri*⟩`}`

to LaTeX, the mapping of these specialized modules is not described in detail. However, in specific use cases the support of these modules might be desired. The following hints might serve as a starting point to implement a transformation of these modules' features to LaTeX.

The HTML Applet, Object, Scripting, and Intrinsic Events modules define markup that introduces scripting facilities to manipulate dynamically the document content. At present this is notably realized through the JavaScript programming language, which is partially integrated with LaTeX by means of the insdljs package.

The HTML Client- and Server-side Image Map modules define markup for interactive and hyperlinked images. This functionality can be potentially realized in LaTeX by means of the Ti*k*Z package.

The HTML Frames and Iframe modules define markup to insert one document into another. This can be realized in LaTeX with the `\input` and/or `\include` commands.

The HTML Style Sheet and Style Attribute modules define markup to integrate layout definitions realized through Cascading Style Sheets (CSS). CSS has its own syntax and description logic — its transformation to LaTeX is a topic all its own, which has been outlined e.g. in [15].

4 An example

Figure 3 shows an example page taken from a bird guide. On the next page, Figure 4 shows a possible coding of this page using HTML, and Figure 5 its corresponding representation in LaTeX.

1

Gannet

Birds of the open ocean, Gannets breed on small islands off the NW coast of Europe. They move away from land after nesting to winter at sea. The young migrate south as far as W Africa. Gannets feed on fish by plunge-diving from 25m. They nest in large, noisy colonies. The nest is a pile of seaweed. A single egg is incubated for 44 days. The young bird is fed by both parents and flies after 90 days.

Size	Larger than any gull
Adult	White, black wing-tips, yellow nape
Juvenile	Grey, gradually becoming white over 5 years
Bill	Dagger-like
In flight	Cigar-shaped with long, narrow, black-tipped wings
Voice	Usually silent, growling "urr" when nesting
Lookalikes	Skuas, Gulls and Terns

Figure 3: An example document, derived from [5]

5 Conclusion

While HTML is increasingly becoming the common document description language for different output media (web, print, e-books, ...), the problem of creating well-typeset documents from HTML is not yet fully solved within the XML ecosystem. The article at hand has introduced a mapping from HTML elements to corresponding LaTeX commands, in order to use the TeX typesetting engine for this task.

With the multitude of existing LaTeX extensions released as packages, almost any HTML description can be ported to LaTeX and typeset according to its original logic. Unfortunately, the use of LaTeX packages often comes with a catch: while many HTML structures can be used recursively (e.g. nested lists or tables), LaTeX packages tend to override existing commands giving them a new meaning (e.g. the newline command is redefined in table environments to end a row). These context-dependent syntax-changes can make a mapping potentially error-prone for deep document structures.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head><title>Gannet</title></head>
  <body>
    <h1>Gannet</h1>
    <p>Birds of the open ocean, Gannets breed on small islands off the <abbr title="northwest"
      >NW</abbr> coast of Europe. They move away from land after nesting to winter at sea. The
      young migrate south as far as <abbr title="west">W</abbr> Africa. Gannets feed on fish by
      plunge-diving from 25<abbr title="meters">m</abbr>. They nest in large, noisy colonies. The
      nest is a pile of seaweed. A single egg is incubated for 44 days. The young bird is fed by
      both parents and flies after 90 days.</p>
    <div><img src="gannet.jpg" alt="Gannet"/></div>
    <table>
      <tr><th>Size</th><td>Larger than any gull</td></tr>
      <tr><th>Adult</th><td>White, black wing-tips, yellow nape</td></tr>
      <tr><th>Juvenile</th><td>Grey, gradually becoming white over 5 years</td></tr>
      <tr><th>Bill</th><td>Dagger-like</td></tr>
      <tr><th>In flight</th><td>Cigar-shaped with long, narrow, black-tipped wings</td></tr>
      <tr><th>Voice</th><td>Usually silent, growling <q>urr</q> when nesting</td></tr>
      <tr><th>Lookalikes</th><td>Skuas, Gulls and Terns</td></tr>
    </table>
  </body>
</html>
```

Figure 4: HTML source, describing the document shown in Figure 3

```
\documentclass{report}
% preamble ...

\begin{document}
\chapter{Gannet}

Birds of the open ocean, Gannets breed on small islands off the \acrshort{NW}coast of Europe. They
move away from land after nesting to winter at sea. The young migrate south as far as \acrshort{W}
Africa. Gannets feed on fish by plunge-diving from 25\acrshort{m}. They nest in large, noisy
colonies. The nest is a pile of seaweed. A single egg is incubated for 44 days. The young bird is
fed by both parents and flies after 90 days.\par

\includegraphics{gannet.jpg}

\begin{longtable}{ll}
\toprule
\bf{}Size       & Larger than any gull \\
\bf{}Adult      & White, black wing-tips, yellow nape \\
\bf{}Juvenile   & Grey, gradually becoming white over 5 years \\
\bf{}Bill       & Dagger-like \\
\bf{}In flight  & Cigar-shaped with long, narrow, black-tipped wings \\
\bf{}Voice      & Usually silent, growling \enquote{urr} when nesting \\
\bf{}Lookalikes & Skuas, Gulls and Terns \\
\bottomrule
\end{longtable}

\end{document}
```

Figure 5: LaTeX output, exported from HTML shown in Figure 4

The mappings introduced in this article have been developed in the context of an XSLT implementation within a TEXML-based workflow, but do not rely on it and can be implemented through other approaches as well. However, the principle of TEXML, to provide a processor that transforms specific TEX commands from a generic XML representation to the TEX format, realizes a separation between the task of format transformation and the task of defining appropriate mappings. This facilitates the definition and adaption of markup correspondences as has e.g. been done by extending the HTML mapping with a third party stylesheet that defines the transformation from MathML to LATEX.

Acknowledgment

The author would like to thank Prof. Dr. Karl-Heinrich Schmidt for giving valuable advice and Gilles Bülow for integrating this document processing workflow into our daily tasks for testing purposes.

References

[1] Daniel Austin, Shane McCarron, Subramanian Peruvemba, Masayasu Ishikawa, and Mark Birbeck. XHTML modularization 1.1 — second edition. W3C recommendation, W3C, July 2010. http://www.w3.org/TR/2010/REC-xhtml-modularization-20100729/.

[2] Robin Berjon, Steve Faulkner, Travis Leithead, Erika Doyle Navara, Edward O'Connor, Silvia Pfeiffer, and Ian Hickson. HTML 5. W3C candidate recommendation, W3C, August 2013. http://www.w3.org/TR/2013/CR-html5-20130806/.

[3] Tim Bray, François Yergeau, C. M. Sperberg-McQueen, Jean Paoli, and Eve Maler. Extensible markup language (XML) 1.0 (fourth edition). W3C recommendation, W3C, August 2006. http://www.w3.org/TR/2006/REC-xml-20060816/.

[4] Markus Gylling, William McCoy, Elika J. Etemad, and Matt Garrish. EPUB content documents 3.0. IDPF recommended specification, IDPF, October 2011. http://www.idpf.org/epub/30/spec/epub30-contentdocs-20111011.html.

[5] Renate Henschel, John Bateman, and Judy Delin. Automatic genre-driven layout generation. In *Proceedings of the 6th "Konferenz zur Verarbeitung natürlicher Sprache" (KONVENS) Conference*, Saarbrücken, September 2002.

[6] Masayasu Ishikawa and Shane McCarron. XHTML 1.1 — module-based XHTML — second edition. W3C recommendation, W3C, November 2010. http://www.w3.org/TR/2010/REC-xhtml11-20101123/.

[7] Ian Jacobs, David Raggett, and Arnaud Le Hors. HTML 4.01 specification. W3C recommendation, W3C, December 1999. http://www.w3.org/TR/1999/REC-html401-19991224/.

[8] Michael Kay. XSL transformations (XSLT) version 2.0. W3C recommendation, W3C, January 2007. http://www.w3.org/TR/2007/REC-xslt20-20070123/.

[9] Sanders Kleinfeld. The case for authoring and producing books in (X)HTML5. In *Proceedings of Balisage: The Markup Conference 2013*, volume 10 of *Balisage Series on Markup Technologies*, Montréal, August 2013.

[10] Donald Ervin Knuth. *The TEXbook*, volume A of *Computers & Typesetting*. Addison-Wesley, March 1986.

[11] Leslie Lamport. *LATEX: A Document Preparation System*. Addison-Wesley, 2nd edition, November 1994.

[12] Douglas Lovell. TEXML: Typesetting XML with TEX. *TUGboat*, 20(3):176–183, September 1999. http://tug.org/TUGboat/tb20-3/tb64love.pdf.

[13] Shane McCarron. XHTML-print — second edition. W3C recommendation, W3C, November 2010. http://www.w3.org/TR/2010/REC-xhtml-print-20101123/.

[14] Oleg Parashchenko. TEXML: Resurrecting TEX in the XML world. *TUGboat*, 28(1):5–10, March 2007. http://tug.org/TUGboat/tb28-1/tb88parashchenko.pdf.

[15] S. Sankar, S. Mahalakshmi, and L. Ganesh. An XML model of CSS3 as an XLATEX-TEXML-HTML5 stylesheet language. *TUGboat*, 32(3):281–284, December 2011. http://tug.org/TUGboat/tb32-3/tb102sankar.pdf.

◇ Frederik R. N. Schlupkothen
University of Wuppertal
Rainer-Gruenter-Str. 21
D-42119 Wuppertal
Germany
schlupko (at) uni-wuppertal dot de