TEX4ht: HTML Production

Eitan M. Gurari
Ohio State University
USA
gurari@cse.ohio-state.edu
http://www.cse.ohio-state.edu/~gurari

Abstract

TEX4ht is a highly configurable system for producing hypertext from TEX-based sources. The system is distributed with a large set of configuration files. The most commonly used configurations are those supporting LATEX inputs and HTML, MathML, OpenOffice, and DocBook targets. The first part of the presentation will describe how the system can be used for different applications.

ConTEXt is a new addition to the style files being supported by TEX4ht. The second part of the presentation will describe the work done to provide TEX4ht configurations for ConTEXt, with the objective of offering an insight into the inner working of TEX4ht.

1 From LATEX to Hypertext

Reports authored in LATEX may be converted into hypertext through the TEX4ht system [1]. The system offers an assortment of basic commands for invoking translations to different target mark-up languages, provides switches for requesting predefined variations to the default configurations, and lets the users tailor configurations of their own.

1.1 Basic Translations

To activate a translation relying on a default configuration, one needs just to invoke an appropriate command and provide it with the LATEX file name. Figure 1 lists a few examples. Most users of the TEX4ht system are probably familiar just with the `htlatex` option. However, the `mzlatex` option seems also to be quite popular.

From the perspective of a user, the process is similar to that employed in requesting a standard translation to DVI or PDF. In such cases, typically the translations are requested through a command named `latex` or `pdflatex`, respectively.

HTML devotes very little support to mathematics, providing only simple superscript and subscript elements. Bitmap representations are offered for mathematical expressions to try to address this shortcoming. Such representations are commonly employed as most users are able to view them in their browsers. Yet, bitmap representations are visually inferior with respect to their surrounding text, as they do not scale in size. In addition, non-visual applications can make little use of these representations.

command	output	comment
`htlatex abc`	`abc.html`	HTML, bitmap math
`xhlatex abc`	`abc.html`	XHTML, bitmap math
`mzlatex abc`	`abc.xml`	XHTML, MathML math
`oolatex abc`	`abc.sxw`	OpenOffice XML (uses MathML math)
`dbmlatex abc`	`abc.xml`	DocBook, MathML math

Figure 1: Requests to compile `abc.tex`.

MathML introduces a markup language for expressing mathematics, in a manner compatible with HTML support of regular text. Currently, not many browsers come with built-in support for MathML. Mozilla is an example of a browser which supports MathML. For Microsoft Internet Explorer, an easily installed plug-in program named MathPlayer offers similar capabilities [2]. Stylesheets are also available to render MathML through XSLT and CSS code [3].

1.2 Available Adjustments

The distribution of TEX4ht provides configurations for default behavior, as well as configurations for achieving alternative outcomes. The latter configurations can be requested by referring to their named options through generalized invocation commands of the following form:

This material is based upon work supported by the National Science Foundation under Award No. IIS-0312487. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author and do not necessarily reflect the views of the National Science Foundation.

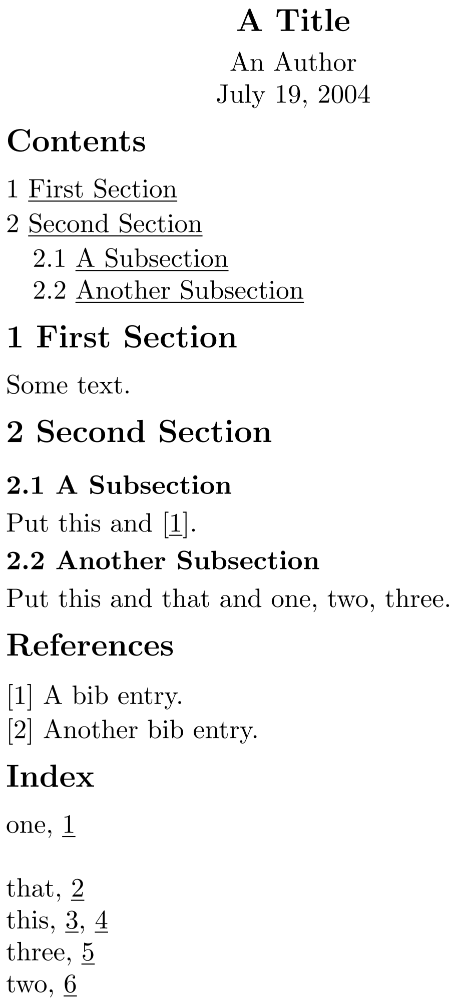
```
\documentclass{article}
   \usepackage{makeidx}
   \makeindex
   \title{A Title}
   \author{An Author}
   \date{July 19, 2004}
\begin{document}
   \maketitle  \tableofcontents

   \section{First Section}
      Some text.

   \section{Second Section}
   \subsection{A Subsection}
         Put \index{this}this
         and \cite{bib-1}.
   \subsection{Another Subsection}
         Put \index{this}this
         and \index{that}that
         and \index{one}one,
         \index{two}two,
         \index{three}three.

   \begin{thebibliography}{99}
     \bibitem{bib-1}
        A bib entry.
     \bibitem{bib-2}
        Another bib entry.
   \end{thebibliography}
   \printindex
\end{document}
```

(a)



A Title

An Author

July 19, 2004

Contents

1 First Section

2 Second Section

2.1 A Subsection

2.2 Another Subsection

1 First Section

Some text.

2 Second Section

2.1 A Subsection

Put this and [1].

2.2 Another Subsection

Put this and that and one, two, three.

References

[1] A bib entry.

[2] Another bib entry.

Index

one, 1

that, 2

this, 3, 4

three, 5

two, 6

(b)

Figure 2: (a) A LaTeX file `source.tex`. (b) A view of the HTML outcome of '`htlatex source`'.

command-name file-name `"html,`*options*`"`

Figure 2(a) lists an example source LaTeX file `source.tex` which requests standard logical structures, including a title segment, sectioning blocks, table of contents, bibliography, and index. A compilation of this file with the command

```
htlatex source
```

produces the default outcome for HTML code. Figure 2(b) shows a possible rendering of this outcome.

A compilation of the same LaTeX file with

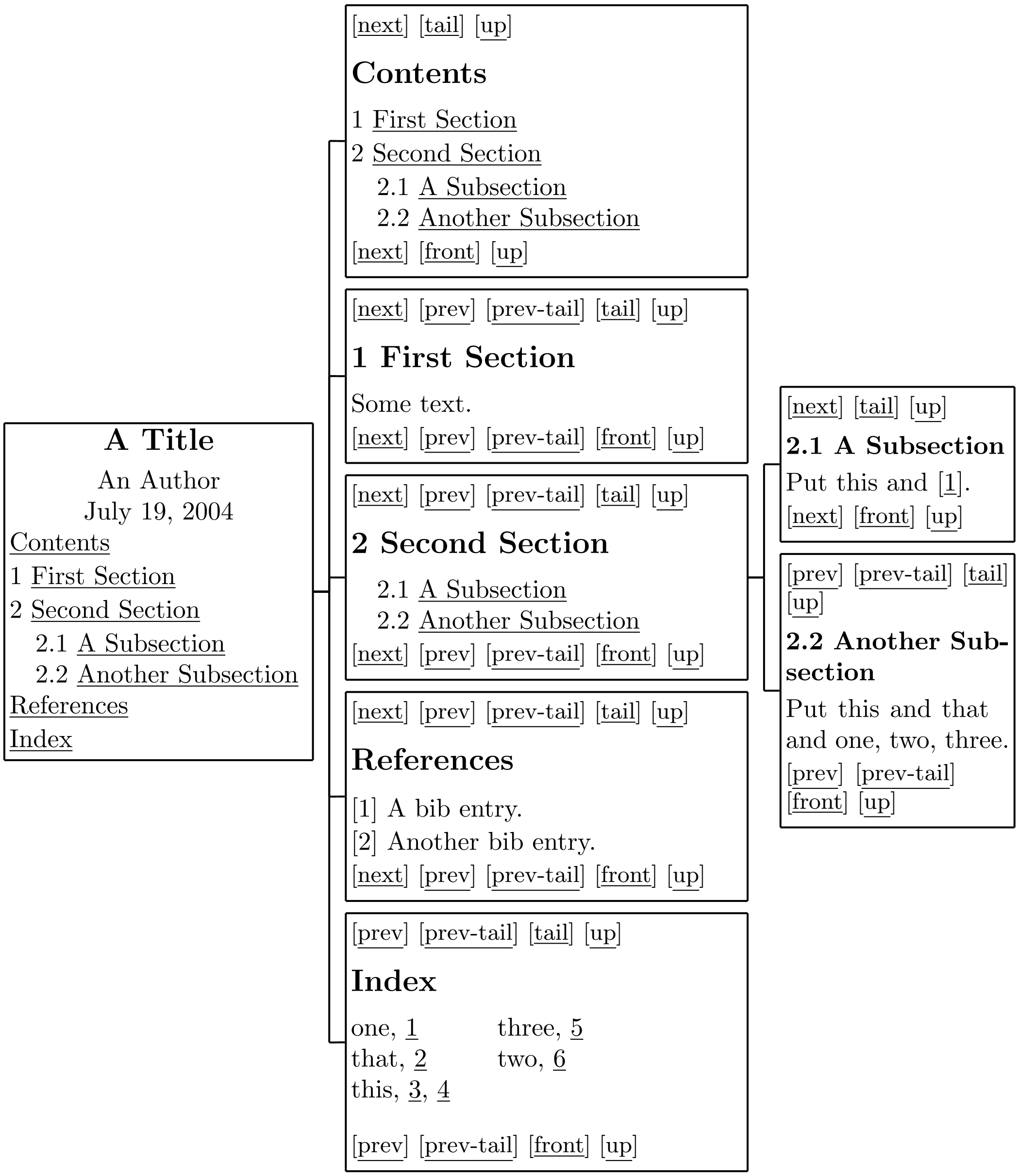
```
htlatex source "html,index=2,3"
```

sets the index in two columns, and partitions the document into web pages based on the sectioning units to a depth of three levels. Figure 3 shows a possible rendering of the different web pages and their hierarchy in a tree structure. The tables of contents enable navigation down the tree levels, and the '`up`' buttons enable navigation in the opposite direction. Navigation between siblings is possible through '`next`' and '`prev`' buttons. For instance, the '`next`' button on the web page of the *Second Section* leads to the web page of the *References.*

A somewhat similar organization of content can be achieved with

```
htlatex source "html,index=2,3,next"
```

Figure 4 shows the result. Here, however, due to the '`next`' option, the '`next`' and '`prev`' navigation buttons assume a different ordering of pages in which the document content is visited sequentially. For instance, under this option the '`next`' button of the root web page leads to the web page of the table of contents. Similarly, the '`next`' button of the web page of *Second Section* leads to the web page of subsection 2.1. On the other hand, the '`next`' button of the web page of subsection 2.2 leads to the web page of the *References.*



A Title

An Author

July 19, 2004

Contents

1 First Section

2 Second Section

2.1 A Subsection

2.2 Another Subsection

References

Index

[next] [tail] [up]

Contents

1 First Section

2 Second Section

2.1 A Subsection

2.2 Another Subsection

[next] [front] [up]

[next] [prev] [prev-tail] [tail] [up]

1 First Section

Some text.

[next] [prev] [prev-tail] [front] [up]

[next] [prev] [prev-tail] [tail] [up]

2 Second Section

2.1 A Subsection

2.2 Another Subsection

[next] [prev] [prev-tail] [front] [up]

[next] [tail] [up]

2.1 A Subsection

Put this and [1].

[next] [front] [up]

[prev] [prev-tail] [tail] [up]

2.2 Another Subsection

Put this and that and one, two, three.

[prev] [prev-tail] [front] [up]

[next] [prev] [prev-tail] [tail] [up]

References

[1] A bib entry.

[2] Another bib entry.

[next] [prev] [prev-tail] [front] [up]

[prev] [prev-tail] [tail] [up]

Index

one, 1

that, 2

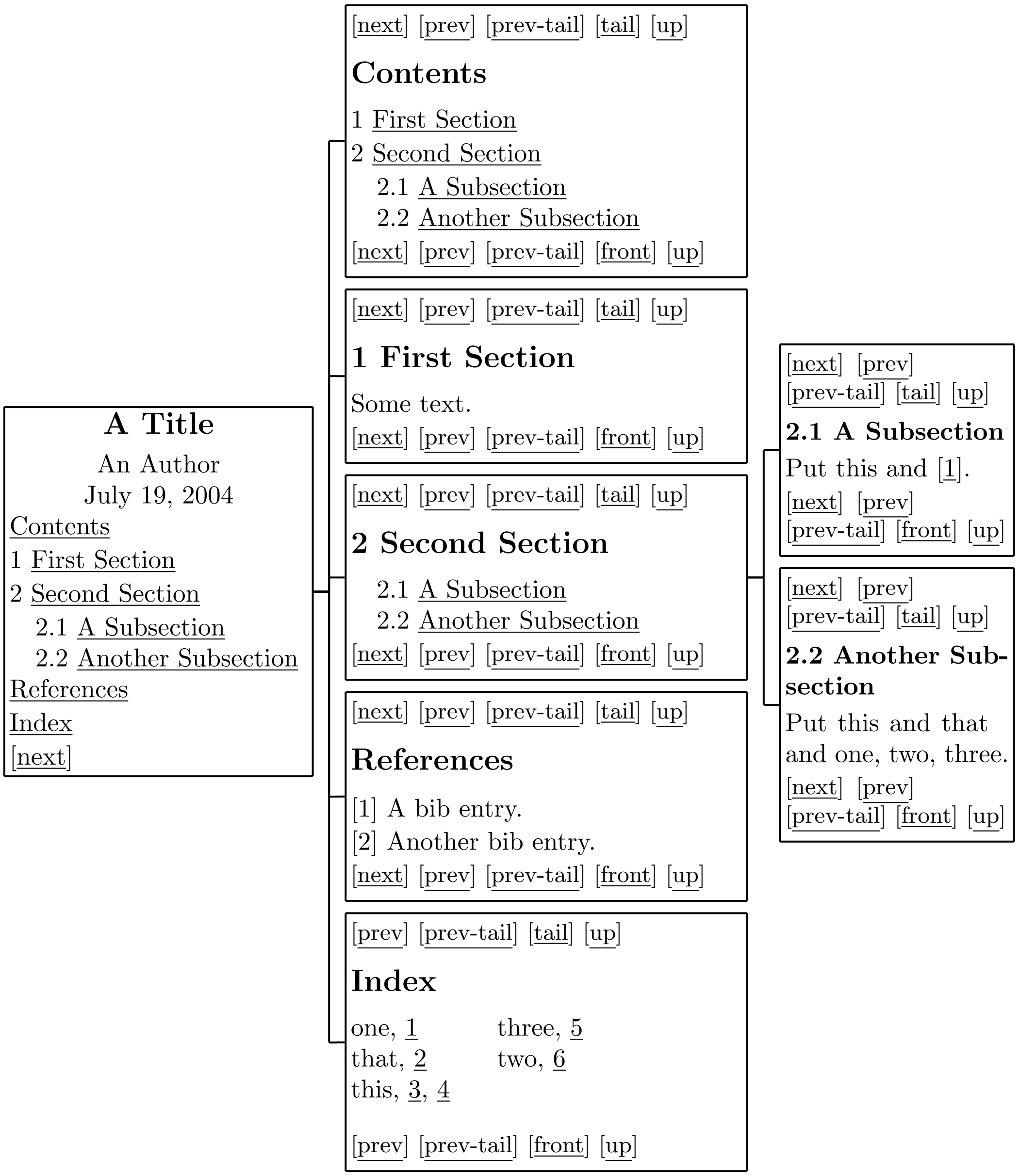
this, 3, 4

three, 5

two, 6

[prev] [prev-tail] [front] [up]

Figure 3: A view of the HTML outcome of '`htlatex source "html,index=2,3"`'. This produces the index in two columns, and separates sections to the third level into their own files.



A Title

An Author

July 19, 2004

Contents

1 First Section

2 Second Section

2.1 A Subsection

2.2 Another Subsection

References

Index

[next]

[next] [prev] [prev-tail] [tail] [up]

Contents

1 First Section

2 Second Section

2.1 A Subsection

2.2 Another Subsection

[next] [prev] [prev-tail] [front] [up]

[next] [prev] [prev-tail] [tail] [up]

1 First Section

Some text.

[next] [prev] [prev-tail] [front] [up]

[next] [prev] [prev-tail] [tail] [up]

2 Second Section

2.1 A Subsection

2.2 Another Subsection

[next] [prev] [prev-tail] [front] [up]

[next] [prev] [prev-tail] [tail] [up]

References

[1] A bib entry.

[2] Another bib entry.

[next] [prev] [prev-tail] [front] [up]

[prev] [prev-tail] [tail] [up]

Index

one, 1	three, 5
that, 2	two, 6
this, 3, 4	

[prev] [prev-tail] [front] [up]

[next] [prev]
[prev-tail] [tail] [up]

2.1 A Subsection

Put this and [1].

[next] [prev]
[prev-tail] [front] [up]

[next] [prev]
[prev-tail] [tail] [up]

2.2 Another Subsection

Put this and that and one, two, three.

[next] [prev]
[prev-tail] [front] [up]

Figure 4: A view of the HTML outcome of '`htlatex source "html,index=2,3,next"`'. Similar to the previous figure, but with sequential navigation, due to the `next` option.

```
/.../texmf/tex/generic/tex4ht/tex4ht.sty
version 2004-05-26-22:19

-- Note -- for automatic sectioning pagination, use
           the command line option ‘1’, ‘2’, or ‘3’

-- Note -- for i-columns index, use the command line
           option ‘index=i’ (e.g., index=2)

-- Note -- for linear crosslinks of pages, use the
           command line option ‘next’

-- Note -- for inline footnotes use
           command line option ‘fn-in’

-- Note -- for content and toc in 2 frames,
           use the command line option ‘frames’

-- Note -- For multi-platform MathML through
           stylesheet transforms, use the command
           line option ‘pmathml’. If css rendering
           is preferred, use ‘pmathml-css’.

-- \TeX4ht{} warning -- If not done so, the index is
to be processed by
  tex ’\def\filename{{source}{idx}{4dx}{ind}}
       \input  idxmake.4ht’
  makeindex -o source.ind source.4dx
instead of
  makeindex -o source.ind source.idx
```

Figure 5: TeX4ht messages recorded in the log.

A few other selected options:

- The ‘`frames`’ option may be used to incorporate a table of contents as a navigation bar for the web pages.
- The ‘`fn-in`’ option asks for footnotes at the end of the web pages, instead of being placed at separate pages.
- The ‘`mouseover`’ option requests pop up messages showing content associated with pointers to footnotes and bibliography entries.

1.3 Log Files: A Source of Information

A compilation of a LaTeX file `source.tex` produces messages that are recorded in a `source.log` file. Some of these messages, though not all of them, are also listed on the user's terminal. The messages depend on the TeX4ht configurations being activated, and contain useful hints, including the available command line options, version indicators, warnings about possible problems, and information about errors encountered. Figure 5 lists a few examples of the messages obtained in compiling the file of Figure 2(a) with the `mzlatex` command.

The listed command line options ‘`3`’, ‘`index=2`’, ‘`next`’, ‘`fn-in`’, and ‘`frames`’ were considered earlier. The ‘`pmathml`’ and ‘`pmathml-css`’ options refer to the stylesheets of [3]. The warning message indicates how indexes are to be compiled.

```
\documentclass{article}
 \def\greeting{Hi}
\begin{document}
 \greeting{} from \LaTeX{}!
\end{document}
```

(a)

```
\Preamble{html}
\begin{document}
 \def\greeting{Hello}
 \def\LaTeX{}{\TeX4ht{}}
\EndPreamble
```

(b)

Figure 6: (a) A LaTeX file `src.tex`. (b) A configuration file `cf.cfg`, changing macros.

LaTeX is a system comprised of a very large set of style files, with new styles being added and old ones being modified periodically. Furthermore, there are numerous ways to represent in hypertext the special properties of the style files. The TeX4ht system is quite often updated to address changes in the LaTeX environment, users' requests for new features, and errors in the implementation.

1.4 User Configurations

A single LaTeX file might be employed by different commands to create a document in an assortment of formats, such as PDF and HTML. Consequently, it is undesirable to explicitly include TeX4ht code in LaTeX sources. Commands of the following form can be used to indirectly load configuration files into compilations (the *cfg-file* is the new piece):

command-name file-name "*cfg-file*,*options*"

An extension `.cfg` is assumed for a configuration file specified without an extension. The configuration file is loaded into the compilation when the start of the LaTeX source body is reached; that is, at ‘`\begin{document}`’. The configuration file must have a structure compatible with the following template:

`\Preamble{`*options*`}`
configurations before the HTML header
`\begin{document}`
configurations within the HTML header
`\EndPreamble`

For instance, Figure 6(a) lists a LaTeX source file whose body is intended to produce the content “Hi from LaTeX!”. Yet, when compiled with the command

```
htlatex src "cf"
```

```
\begin{itemize}
  \item First item
  \item Second item
\end{itemize}
```

(a)

```
<ul class="itemize1">
   <li class="itemize">
      First item
   </li>
   <li class="itemize">
      Second item
   </li>
</ul>
```

(b)

```
\Preamble{html}
\begin{document}
  \Css { ul.itemize1 {
    color : red ;
    background-color : yellow;
    font-weight: bold ;
    font-size : 150\%
  }}
  \Css { li {
    border : black 1px solid;
    margin : 2em ;
    text-align : center
  }}
\EndPreamble
```

(c)

Figure 7: (a) LaTeX fragment. (b) Corresponding HTML code. (c) Possible CSS configurations.

the outcome is “Hello from TeX4ht!” given the configuration file `cf.cfg` listed in Figure 6(b). This example illustrates, rather dramatically, the idea and potential of configuration files. However, configuration files are typically used to tailor mark-up for the content, not to actually *change* the content!

1.5 Touch-Up With CSS

According to its definition, “Cascading Style Sheets (CSS) is a simple mechanism for adding style (e.g., fonts, colors, spacing) to Web documents” [4]. Accordingly, TeX4ht provides a `\Css{...}` instruction for incorporating CSS code into target files.

Figure 7(a) lists a sample LaTeX source fragment. When compiled for HTML output, the code produced is as listed in Figure 7(b). The configuration file of Figure 7(c) can be introduced into the compilation to associate the given CSS decorations with the HTML code.

1.6 Changing HTML Configurations

TeX4ht indirectly seeds hooks within the LaTeX constructs and associates default configurations with the constructs through the hooks. Users can change these configurations, but typically should do this with a good understanding of LaTeX programming, HTML, and TeX4ht.

Hints as to how the default configurations can be modified may be seen through the ‘`info`’ command line option. The hints are recorded within the log files of the compilations. For instance, the command

```
htlatex source "html,info"
```

shows hints like this within `source.log`:

- Configure environments `\begin{`*name*`} ... \end{`*name*`}` with `\ConfigureEnv{`*name*`} {...} {...} {...} {...}`
- Configure lists `\begin{`*name*`}\item ... \item ... \end{`*name*`}` with `\ConfigureList {`*name*`}{...}{...}{...}{...}`

Figure 8(a) exhibits a possible use of the above instructions for configuring typical LaTeX sources, as shown in Figure 8(b). The outcome is listed in Figure 8(c).

1.7 Beware of Errors: Validate

LaTeX is forgiving of different kinds of misuses of the language. In addition, TeX4ht is not configured for all features of LaTeX and their possible interactions. In contrast, hypertext markup languages impose strict requirements on their use. Consequently, translations are not immune to errors and users are therefore encouraged to validate the output files.

Validators can be invoked via constructs similar to ‘`.html` *utility* `%%1.html`’ in the system environment file `tex4ht.env`.

2 Configuring TeX4ht for ConTeXt

ConTeXt is a macro package offering high-level constructs for expressing logical units of documents [5]. The remainder of this report describes what it took to introduce support for ConTeXt in TeX4ht. The underlying ideas are similar to those employed to support LaTeX.

2.1 Getting Background Information

TeX4ht processes a source file by indirectly modifying the style files in use, invoking the native compiler to translate the source file into DVI code and then processing the DVI output into hypertext markup. In the case of ConTeXt, ‘`texexec` *filename*’ is the basic command to output DVI code.

```
\ConfigureEnv{titlepage}
  {\ifvmode \IgnorePar\fi
   \EndP
   \HCode{<h1>}\IgnorePar }
  {\ifvmode \IgnorePar\fi
   \EndP \HCode{</h1>}}
  {} {}
\ConfigureList{enumerate}
  {\HCode{<div>}}
  {\HCode{</div>}}
  {\HCode{<span
          class="mark">}}
  {\HCode{</span>} }
```

(a)

```
\begin{titlepage}
  Some Title
\end{titlepage}
\begin{enumerate}
  \item First item
  \item Second item
\end{enumerate}
```

(b)

```
<h1> Some Title </h1>
<div><span class="mark">
1. </span> First item
   <span class="mark">
2. </span> Second item </div>
```

(c)

Figure 8: (a) TeX4ht configurations changing the HTML output for the `titlepage` and `enumerate` environments. (b) LaTeX source. (c) HTML outcome.

The modification of the style files consisted of indirectly seeding hooks into the files and providing default configurations to the seeds. To achieve this end, simple sample files were needed for experimenting with the features under consideration and learning the issues involved. The files had to be minimal in size and address the different issues in isolation.

Berend de Boer offers a rich assortment of simple source ConTeXt files [6]. These files turned out to be very helpful in the development of TeX4ht support for ConTeXt.

2.2 Proof of Concept

To provide support for a new package, TeX4ht must find a way to indirectly access the different features introduced by the package. The first challenge was to determine whether TeX4ht can deal with the simplest ConTeXt source files.

```
\starttext
Hello world.
\stoptext
```

(a)

```
\input tex4ht.sty
\Preamble{xhtml}
\EndPreamble
\starttext
Hello world.
\stoptext
```

(b)

Figure 9: (a) The simplest ConTeXt file. (b) Explicit request for TeX4ht configuration within the ConTeXt file.

To answer this question, the `hello.tex` source of Figure 9(a) was compiled for DVI output with the command '`texexec hello`'. The successful compilation ensured that ConTeXt installed correctly and that the source file was correct. The next stage consisted of creating a similar file `hello4ht.tex` that explicitly loaded the core TeX4ht configurations into the compilation. This modified file is shown in Figure 9(b).

The modified file was similarly compiled with the command '`texexec hello4ht`' to produce DVI output. Then the sequence of commands '`tex4ht hello4ht`' and '`t4ht hello4ht`' post-processed the DVI output into HTML format. The compilation into the DVI target complained along the way about a few errors. Similarly, the post-processing created an imperfect HTML file, with extra text scattered around.

The above problems called for a few corrections to the core TeX4ht configurations. In addition, they required the tailoring of a nucleus of a TeX4ht configuration file `context.4ht` for ConTeXt.

The configuration file incrementally grew in size as the different features of ConTeXt were treated for TeX4ht support. Eventually, all the HTML code was transferred into a configuration file `html4.4ht` dedicated to handling HTML code, and `context.4ht` contained just the code for seeding ConTeXt hooks.

2.3 Setting an Invocation Script

A desirable objective of TeX4ht is to leave the user source file and the ConTeXt style files untouched. In the case of ConTeXt, a new `htcontext` command was introduced to invoke the following script.

```
\def\complexstartsmaller[#1]%
  {\par \bgroup ...
   \advance\leftskip ...
   \advance\rightskip ...}
\def\stopsmaller{\par \egroup}
```

(a)

```
\let\o:complexstartsmaller: =
           \complexstartsmaller
\def\complexstartsmaller[#1]{%
   \o:complexstartsmaller:[#1]%
   \a:narrower\bgroup
   \aftergroup\b:narrower
   \aftergroup\egroup }
\NewConfigure{narrower}{2}
```

(b)

```
\Configure{narrower}
   {\ifvmode\IgnorePar\fi \EndP \HCode{<div class="narrower">}}
   {\ifvmode\IgnorePar\fi \EndP \HCode{</div>}}
\Css{div.narrower {margin-left:2em; margin-right:2em;}}
```

(c)

Figure 10: (a) ConTeXt's `\complexstartsmaller` macro. (b) TeX4ht hooks. (c) HTML configuration.

```
texexec \
  --arg="opt-arg=configuration-options" \
  --use=tex4ht ConTeXt-options filename
tex4ht filename tex4ht-options
t4ht filename t4ht-options
```

The `texexec` command line loads the TeX4ht configurations, including the file `context.4ht`, at the `\starttext` instruction of the source file. The `\starttext` instruction marks the end of the preamble of the document and the start of the body. Consequently, the TeX4ht configurations have the last word on how the environment will look for the compilations into DVI.

2.4 Planting and Configuring Hooks

Planting hooks indirectly into a package's macros requires a deep understanding of the implementation of the macros. For many features, acquiring such knowledge is not an easy task. Experiments with simple source files that use these features can provide very helpful hints. Still, the job is often tedious and time consuming.

Figure 10(b) illustrates how TeX4ht hooks are indirectly introduced, within the `context.4ht` file, into the ConTeXt macro `\complexstartsmaller`. This macro is defined in the style file `core-spa.tex` of ConTeXt — its outline is shown in Figure 10(a). The implementation takes advantage of having the `context.4ht` file loaded at the `\starttext` instruction, while `core-spa.tex` is loaded earlier.

Figure 10(c) shows the HTML configurations to be associated with the hooks in the default setting.

2.5 Observations

ConTeXt is a TeX environment very rich in features. The work described in this report relates to the core ConTeXt features discussed in [6]. Additional configurations will be provided in response to requests from users of the system.

The following are a few of the hardships encountered in preparing TeX4ht configurations for ConTeXt.

- ConTeXt is written in Dutch. Not knowing the language makes it difficult to follow the meaning of commands.
- Having a limited understanding of ConTeXt, too much time was spent on brute force experimentations and tracing of computations.
- General purpose environments such as '`\begin {`*env*`} ...\end{`*env*`}`' in LaTeX are very rewarding. They require very few hooks and cover large sets of commands. ConTeXt offers similar environments through hidden definitions to macros of the form '`\??`*env*'.
- Lack of a clear semantics makes it difficult to provide intelligent configurations (this seemed to be the case for enumerated versus description lists).
- Hooks at different levels of grouping make it difficult to communicate information between the hooks. For instance, the `\@@somedefinitie` macro apparently forces this type of approach.

The ConTeXt system has been created to help produce good looking documents with well-specified page formats, often in PDF format. In this respect it has achieved outstanding results. Hypertext seems to offer a large array of additional opportunities for this system.

2.6 Acknowledgment

I am grateful to Bob Kerstetter for initially requesting TEX4ht configurations for ConTEXt and for arranging help to get me started with ConTEXt. I am indebted to Patrick Gundlach for his considerable effort to install ConTEXt on my platform. I would like to thank Hans Hagen for his input, and Karl Berry for editing the report.

References

[1] Eitan M. Gurari, *TEX4ht: LATEX and TEX for Hypertext*, `http://www.cse.ohio-state.edu/~gurari/TeX4ht/`.

[2] Design Science, *MathPlayer*, `http://www.dessci.com/en/products/mathplayer/default.htm`.

[3] David Carlisle, *XSLT stylesheets for MathML*, `http://www.w3.org/Math/XSL/Overview-tech.html`.

[4] *Cascading Style Sheets (CSS)*, `http://www.w3.org/Style/CSS/`.

[5] Ton Otten and Hans Hagen, *ConTEXt: An excursion*, Pragma Ade, `http://pragma-ade.nl/general/manuals/mp-cb-en.pdf`.

[6] Berend de Boer, *LATEX in proper ConTEXt*, `http://www.berenddeboer.net/tex/LaTeX2ConTeXt.pdf`, July 2003.