The step4ht TeX4ht package*

Peter Wilson
Catholic University of America
Now at peter.r.wilson@boeing.com

2000/01/20

Abstract

The step4ht package, in conjunction with TeX4ht can be used to convert LaTeX STEP documents into HTML tagged documents.

Contents

1 Introduction

The step4ht package can be used in conjunction with the TeX4ht system to convert LaTeX ISO 10303 (STEP) documents into HTML tagged documents.

Section 2 describes the package and commented source code for the package is in Section 3.

*This file has version number v0.1, last revised 2000/01/20.

This manual is typeset according to the conventions of the LaTeX docstrip utility which enables the automatic extraction of the LaTeX macro source files [GMS94].

1.1 Acknowledgement

Development of the step4ht package would not have been possible without the help and expertise of Eitan Gurari, and in particular his willingness to put up with the many questions I asked.

2 The step4ht package

The TeX4ht system has been developed by Eitan Gurari (see Chapter 4 and Appendix B in [GR99]). It is a general purpose conversion system to convert LaTeX tagged documents into HTML (or other *ML) tagged documents. TeX4ht can be obtained from `http://www.cis.ohio-state.edu/~gurari/TeX4ht/mn.html`. The step4ht package is not guaranteed to work with versions of TeX4ht earlier than mid-January 2000. At the time of writing, the default TeX4ht distribution was dated mid-1999. The latest version of TeX4ht is obtainable from `http://www.cis.ohio-state.edu/~gurari/TeX4ht/bugfixes.html`.

To use the step4ht package, just process the LaTeX document as you would any other under TeX4ht. That is, either call the `ht` script on a document starting like:

```
\documentclass[...]{isov2}
\usepackage[...]{tex4ht}
\usepackage{isov13}
...
```

or call the `htlatex` script on a document without the `\usepackage[...]{tex4ht}` line.

3 The package code

The following code is based on html0.4ht, html32.4ht and html4.4ht, all written by Eitan Gurari, together with code in the iso4ht package.

Most of the necessary work already exists in latex.4ht, which sets up the LaTeX kernel code and the iso4ht package which provides the setup and configuration for the iso class, together with html0.4ht, html32.4ht and html4.4ht provide the HTML option-related configurations for the kernel code. The HTML option-related configurations are embedded in stepv13.4ht instead of being supplied as seperate files.

Announce the name and version of the package.

```
1 ⟨*usc⟩
2 \typeout{[stepv13.4ht 2000/01/20 version v0.1]}
3
```

3.1 Setup and hooks

The first major part of the code deals with setting up for configuring the LaTeX commands and environments, which forms the second major portion of the code.

Setting up may involve adding hooks into commands, either by redefining them or, in simpler cases prepending and/or appending code before and/or after the original code. It can also involve specifying that commands are configurable.

3.1.1 The STEP cover page

The `\STEPcover` command is implemented as a `picture` environment. LaTeX pictures are usually presented as `.gif` images in TeX4ht, so some major surgery is done to avoid pictures in this case.

First, we redefine the commands that correspond to the `\STEPcover` picture placement commands.

`\wg` The Working Group.

```
\renewcommand{\wg}[1]{\def\@wg{/#1}}
```

`\oldwg` The old Working Group.

```
\renewcommand{\oldwg}[1]{\def\@oldwg{/#1}}
```

`\@docnumber` `\docnumber` `\donumber` The document number, where `\donumber` has hooks and typesets the number.

```
\renewcommand{\@docnumber}{/N}
\renewcommand{\docnumber}[1]{\def\@docnumber{/N#1}}
\def\donumber{\a:donumber ISO TC 184/SC4\@wg\@docnumber \b:donumber}
\NewConfigure{donumber}{2}

```

`\@docdate` `\docdate` `\dodocdate` The document date, where `dodocdate` has hooks and typesets the date.

```
\renewcommand{\@docdate}{Date: }
\renewcommand{\docdate}[1]{\def\@docdate{\textbf{Date: #1}}}
\def\dodocdate{\a:dodocdate \@docdate \b:dodocdate}
\NewConfigure{dodocdate}{2}

```

`\@olddocnumber` `\olddocnumber` `\doolddocnumber` The previous document number, where `\dooldnumber` has hooks and typesets the number.

```
\renewcommand{\@olddocnumber}{/N}
\renewcommand{\olddocnumber}[1]{\def\@olddocnumber{/N#1}}
\def\dooldnumber{\a:dooldnumber Supersedes ISO TC 184/SC4\@oldwg\@olddocnumber \b:dooldnumber}
\NewConfigure{dooldnumber}{2}

```

`\dopartno` Typeset the Part number (and ballot cycle).

```
\newcommand{\dopartno}{%
  \ifnum\value{b@cyc} < 2\relax
    {\bf ISO\thest@tus\ 10303-\thespartno}
  \else
    {\bf ISO\thest@tus\ 10303-\thespartno.\theb@cyc}
  \fi}
\pend:def\dopartno{\a:dopartno}
```

```
\append:def\dopartno{\b:dopartno}
\NewConfigure{dopartno}{2}
```

\doptitle Typeset the Part title.

```
\newcommand{\doptitle}{%
  {\bf \st@pn@me : \thes@ries : \thed@ctitle}
}
\pend:def\doptitle{\a:doptitle}
\append:def\doptitle{\b:doptitle}
\NewConfigure{doptitle}{2}
```

\cpnotice Typeset the title COPYRIGHT NOTICE.

```
\newcommand{\cpnotice}{\a:cpnotice COPYRIGHT NOTICE \b:cpnotice}
\NewConfigure{cpnotice}{2}
```

\@abstract \abstract \doabstract The abstract, where \doabstract has hooks and typesets the abstract.

```
\renewcommand{\@abstract}{\textbf{ABSTRACT}: }
\renewcommand{\abstract}[1]{\def\@abstract{\textbf{ABSTRACT}: #1}}
\def\doabstract{\a:doabstract \@abstract \b:doabstract}
\NewConfigure{doabstract}{2}
```

\@keywords \keywords \dokeywords The KEYWORDS, where \dokeywords has hooks and typesets the keywords.

```
\renewcommand{\@keywords}{\textbf{KEYWORDS}: }
\renewcommand{\keywords}[1]{\def\@keywords{\textbf{KEYWORDS}: #1}}
\def\dokeywords{\a:dokeywords \@keywords \b:dokeywords}
\NewConfigure{dokeywords}{2}
```

\@comread \comread \docomread The COMMENTS TO READERS, where \docomread has hooks and typesets the comments.

```
\renewcommand{\@comread}{\textbf{COMMENTS TO READER}: }
\renewcommand{\comread}[1]{\def\@comread{\textbf{COMMENTS TO READER}: #1}}
\def\docomread{\a:docomread \@comread \b:docomread}
\NewConfigure{docomread}{2}
```

\owner Contact information for the project leader.

\address \telephone \fax \email

```
\renewcommand{\owner}[1]{\def\@owner{#1}}
\renewcommand{\address}[1]{\def\@address{#1}}
\renewcommand{\telephone}[1]{\def\@telephone{#1}}
\renewcommand{\fax}[1]{\def\@fax{#1}}
\renewcommand{\email}[1]{\def\@email{#1}}
```

\altowner Contact information for the document editor.

\altaddress \alttelephone \altfax \altemail

```
\renewcommand{\altowner}[1]{\def\@altowner{#1}}
\renewcommand{\altaddress}[1]{\def\@altaddress{#1}}
\renewcommand{\alttelephone}[1]{\def\@alttelephone{#1}}
```

```
\renewcommand{\altfax}[1]{\def\@altfax{#1}}
\renewcommand{\altemail}[1]{\def\@altemail{#1}}
```

\STEPcover

The \STEPcover command is redefined to add hooks at the start and end.

```
\renewcommand{\STEPcover}[1]{
\a:STEPcover
#1
\drawcoversheet
\b:STEPcover
}
\NewConfigure{STEPcover}{2}
```

\drawcoversheet

\drawcoversheet is redefined to typeset the cover instead of drawing it.

```
\renewcommand{\drawcoversheet}{
\donumber \\
\dodocdate \\
\dooldnumber \\
\dopartno \\
\doptitle \\

\ifc@pyrightopt
  \ifisst@ndard   \input{bpfsX} \fi  % IS
  \iffdisst@ndard \input{bpfs2} \fi  % FDIS
  \ifdisst@ndard  \input{bpfs3} \fi  % DIS
  \ifcdst@ndard   \input{bpfs4} \fi  % CD
  \ifwdst@ndard   \input{bpfs4} \fi  % WD
  \ift@chrep      \input{bpfsX} \fi  % Tech Report
\fi

\doabstract \\
\dokeywords \\
\docomread \\
\begin{tabular}{|l|p{0.3\textwidth}|p{0.3\textwidth}|} \hline
    & Project Leader & Project Editor \\ \hline
Name & \@owner & \@altowner \\
Address & \@address & \@altaddress \\
Telephone & \@telephone & \@alttelephone \\
TeleFacsimile & \@fax & \@altfax \\
Email & \@email & \@altemail  \\ \hline
\end{tabular}
```

And, as before, clear out the cover commands which are no longer needed.

```
  \undef@covercmds
}
```

3.1.2 Miscellaneous

\@presteptitle
\stepparttitle

The \stepparttitle is redefined to use the \titleclause* command as defined in the iso4ht package. The \@presteptitle command is just a helper. The configuration of \titleclause* is defined in iso4ht.

```
\newcommand{\@presteptitle}{\scivm@in \stepc@mp \thisp@rtno{\thespartno}}
\renewcommand{\stepparttitle}[1]{%
  \gdef\thestepparttitle{{\@presteptitle \sptitle{#1}}}
  \titleclause*{\thestepparttitle}
}

```

3.2 Configuration

All, or nearly all, configurations depend on the HTML level option chosen. Typically, html0 results in empty or null values of the hooks; the \NewConfigure command provides empty configurations. html32 has some simple hook values, while html4 are the most complex.

3.2.1 STEP cover page

\STEPcover
\donumber
\dodocdate
\dooldnumber
\dopartno
\doptitle
\cpnotice
\doabstract
\dokeywords
\docomread

```
\:CheckOption{0.0}
\if:Option
  %%%% html0.0

\else
  \:CheckOption{3.2}
  \if:Option
  %%%% html3.2
    \Configure{STEPcover}{\HCode{<hr>}}{\HCode{<hr>}}
    \Configure{donumber}{\HCode{<h3>}}{\HCode{</h3>}}
    \Configure{dodocdate}{\HCode{<h5>}}{\HCode{</h5>}}
    \Configure{dooldnumber}{\HCode{<h4>}}{\HCode{</h4>}}
    \Configure{dopartno}{\HCode{<h4>}}{}
    \Configure{doptitle}{}{\HCode{</h4>}}
    \Configure{cpnotice}{\HCode{<h5>}}{\HCode{</h5>}}
    \Configure{doabstract}{\HCode{<blockquote>}}{\HCode{</blockquote>}}
    \Configure{dokeywords}{\HCode{<blockquote>}}{\HCode{</blockquote>}}
    \Configure{docomread}{\HCode{<blockquote>}}{\HCode{</blockquote>}}

  \else
  %%%% html4.0
    \Configure{STEPcover}{\HCode{<hr>}}{\HCode{<hr>}}
    \Configure{STEPcover}{\HCode{<hr>}}{\HCode{<hr>}}
    \Configure{donumber}{\HCode{<h3>}}{\HCode{</h3>}}
    \Configure{dodocdate}{\HCode{<h5>}}{\HCode{</h5>}}
    \Configure{dooldnumber}{\HCode{<h4>}}{\HCode{</h4>}}
    \Configure{dopartno}{\HCode{<h4>}}{}
```

```
    \Configure{doptitle}{}{\HCode{</h4>}}
    \Configure{cpnotice}{\HCode{<h5>}}{\HCode{</h5>}}
    \Configure{doabstract}{\HCode{<blockquote>}}{\HCode{</blockquote>}}
    \Configure{dokeywords}{\HCode{<blockquote>}}{\HCode{</blockquote>}}
    \Configure{docomread}{\HCode{<blockquote>}}{\HCode{</blockquote>}}

  \fi
\fi
```

3.2.2 Miscellaneous

`\start:env` `\end:env` Currently these are defined in the `html*.4ht` files but they might dissapear in future, so make sure they are defined.

```
\:CheckOption{0.0}
\if:Option
  %%%% html0.0
  \def\start:env#1{}
  \def\end:env{}
\else
  \CheckOption{3.2}
  \if:Option
  %%%% html3.2
    \def\start:env#1{\IgnorePar\HCode{<\tbl:XV{#1}><tr><td\Hnewline}}
    \def\end:env{\IgnorePar \end:TTT\ShowPar}
  \else
  %%%% html4
    \def\start:env#1{\IgnorePar\EndP\HCode{<div class="#1"><\tbl:XV{#1}><tr
          class="#1"><td\Hnewline class="#1">}}
    \def\end:env{\IgnorePar \end:TTT\HCode{</div>}\ShowPar}
  \fi
\fi
```

`espec` `fspec` `rspec` `sspec` `tspec` `dtext` `attrlist` `fproplist` `iproplist` `enumlist` `arglist` `majorsublist` These are all simple environments, except for `majorsublist` which is a wrapper for `itemize`.

```
\:CheckOption{0.0}
\if:Option
  %%%% html0.0
  \ConfigureEnv{espec}{}{}{}{}
  \ConfigureEnv{fspec}{}{}{}{}
  \ConfigureEnv{rspec}{}{}{}{}
  \ConfigureEnv{sspec}{}{}{}{}
  \ConfigureEnv{tspec}{}{}{}{}
  \ConfigureEnv{dtext}{}{}{}{}
  \ConfigureEnv{attrlist}{}{}{}{}
  \ConfigureEnv{fproplist}{}{}{}{}
  \ConfigureEnv{iproplist}{}{}{}{}
  \ConfigureEnv{enumlist}{}{}{}{}
```

```
  \ConfigureEnv{arglist}{}{}{}{}
  \ConfigureEnv{majorsublist}{}{}{}{}

\else
  \:CheckOption{3.2}
  \if:Option
  %%%% html3.2
    \ConfigureEnv{espec}{\start:env{espec}}{\end:env}{}{}
    \ConfigureEnv{fspec}{\start:env{fspec}}{\end:env}{}{}
    \ConfigureEnv{rspec}{\start:env{rspec}}{\end:env}{}{}
    \ConfigureEnv{sspec}{\start:env{sspec}}{\end:env}{}{}
    \ConfigureEnv{tspec}{\start:env{tspec}}{\end:env}{}{}
    \ConfigureEnv{dtext}{\start:env{dtext}}{\end:env}{}{}
    \ConfigureEnv{attrlist}{\start:env{attrlist}}{\end:env}{}{}
    \ConfigureEnv{fproplist}{\start:env{fproplist}}{\end:env}{}{}
    \ConfigureEnv{iproplist}{\start:env{iproplist}}{\end:env}{}{}
    \ConfigureEnv{enumlist}{\start:env{enumlist}}{\end:env}{}{}
    \ConfigureEnv{arglist}{\start:env{arglist}}{\end:env}{}{}
    \ConfigureEnv{majorsublist}{\start:env{majorsublist}}{\end:env}{}{}

  \else
  %%%% html4.0
    \ConfigureEnv{espec}{\start:env{espec}}{\end:env}{}{}
    \ConfigureEnv{fspec}{\start:env{fspec}}{\end:env}{}{}
    \ConfigureEnv{rspec}{\start:env{rspec}}{\end:env}{}{}
    \ConfigureEnv{sspec}{\start:env{sspec}}{\end:env}{}{}
    \ConfigureEnv{tspec}{\start:env{tspec}}{\end:env}{}{}
    \ConfigureEnv{dtext}{\start:env{dtext}}{\end:env}{}{}
    \ConfigureEnv{attrlist}{\start:env{attrlist}}{\end:env}{}{}
    \ConfigureEnv{fproplist}{\start:env{fproplist}}{\end:env}{}{}
    \ConfigureEnv{iproplist}{\start:env{iproplist}}{\end:env}{}{}
    \ConfigureEnv{enumlist}{\start:env{enumlist}}{\end:env}{}{}
    \ConfigureEnv{arglist}{\start:env{arglist}}{\end:env}{}{}
    \ConfigureEnv{majorsublist}{\start:env{majorsublist}}{\end:env}{}{}

  \fi
\fi
```

`expdesc` This is a list environment, similar to the `description` environment, except that there is a colon after the label. It is called from within the `...list` environments after a local heading.

```
\:CheckOption{0.0}
\if:Option
  %%%% html0.0
  \ConfigureList{expdesc}{}{}{}{}

\else
  \:CheckOption{3.2}
  \if:Option
```

```
  %%%% html3.2
    \ConfigureList{expdesc}%
      {\HCode{<dl>}}
      {\HCode{</dl>}\ShowPar}
      {\HCode{<dt>\bgroup \bf}
      {: \egroup\HCode{<dd\Hnewline>}}

  \else
  %%%% html4.0
    \ConfigureList{expdesc}%
      {\EndP\HCode{<dl class="expdesc">}\let\end:itm=\empty}
      {\EndP\HCode{</dd></dl>}\ShowPar}
      {\end:itm\def\end:itm{\EndP\Tg</dd>}
        \HCode{<dt class="expdesc">}\bgroup \bf}
      {: \egroup\EndP\HCode{</dt><dd\Hnewline class="expdesc">}}

  \fi
\fi


\endinput

```

The end of the package

⟨/usc⟩

3.3 Observations

There are three main aspects to developing TeX4ht code and configurations for a new package:

1. Finding out what hooks and configurations are already available.
2. Determining what additional hooks, and where they should be put, for the new package.
3. Configuring all the hooks.

TeX4ht has added many hooks to the LaTeX kernel and packages based on kernel code inherit those hooks. I found it advisable to run a test document with the new package(s) through TeX4ht to see what the result looked like. Depending on the package it may be that the inherited hooks and configuration are sufficient and nothing needs to be done.

For this particular package, only a few new hooks were required, but it was necessary in some cases to drastically revamp some of the package code — most noticeably for the `\STEPcover` command. Actually, as I am also the author of the step package I did have the luxury of being able to change some of the internal package code to make a better match between the LaTeX typesetting view of the world and the TeX4ht world view.

There are other packages that, in their turn, use the step package and it turned out that they required no new hooks or changes to the existing configurations.

The rest of this section is concerned with item 2, namely adding hooks.

3.3.1 Colon is a letter

Within the TeX4ht 'environment', that is the `*.4ht` files, the colon character (`:`) acts as a letter in a similar manner as the at character (`@`) does in class and package files; `@` is also treated as a letter in the environment. Normally, something like `\start:env{myenv}` would be treated as the command `\start` followed by the text `:env{myenv}`. In the TeX4ht environment it is the command `\start:env` with the argument `{myenv}`. If any commands like this occur in the preamble to a LaTeX document, then they must be surrounded by `\makecolonletter` and `\restorecolon`, which may be defined as:[1]

```
\chardef\oldcolon=\thecatcode`\:
\newcommand{\makecolonletter}{\catcode`\:11\relax}
\newcommand{\restorecolon}{\catcode`\:=\oldcolon\relax}
```

The fact that command names can include a colon means that you have to be careful in code that includes any colon characters. In LaTeX, code like like `{footnote \thefootnote:}` will print the footnote number immediately followed by a colon (e.g., `footnote 3:`). In the TeX4ht environment you are more likely to get an error message saying that `\thefootnote:` is undefined! Instead, this needs to be coded as `{footnote \thefootnote :}`, so that the `\thefootnote` command is ended by the space before the colon.

3.3.2 Adding code and hooks

A LaTeX idiom for adding code at the start and/or end of an existing macro which takes no arguments called, say `\foo`, is:

```
\let\oldfoo\foo
\renewcommand{\foo}{new-start-code \oldfoo new-end-code}
```

Similarly for commands `\baz` and `\biz` which take one and two arguments respectively:

```
\let\oldbaz\baz
\renewcommand{\baz}[1]{new-start-code \oldbaz{#1} new-end-code}
\let\oldbiz\biz
\renewcommand{\biz}[2]{new-start-code \oldbiz{#1}{#2} new-end-code}
```

[1] Note that a package may redefine the category code for the colon, which is why the old value is saved.

As these kinds of redefinitions are a common occurrence `tex4ht.sty` provides commands that encapsulate the above idiom. These are `\pend:def\foo{new-start-code}` and `\append:def\foo{new-end-code}` for when `\foo` is a macro without arguments, and there are similar commands for prepending and appending to macros with up to three arguments. Repeating and extending the LaTeX example, in the TeX4ht environment it could be coded as:

```
\pend:def\foo{new-start-code}    % \foo has no arguments
\append:def\foo{new-end-code}
\pend:defI\baz{new-start-code}   % \baz has one argument
\append:defI\baz{new-end-code}
\pend:defII\biz{new-start-code}  % \biz has two arguments
\append:defII\biz{new-end-code}
\pend:defIII\boz{new-start-code}  % \boz has three arguments
\append:defIII\boz{new-end-code}
```

The definition of a hook in a macro called, say `\buz`, takes the form `\X:buz` where X is a single letter. For example, adding a configurable hook at the start and end of the macro `\baz` can be done like this:

```
\pend:defI\baz{\a:baz}      % hook at start
\append:defI\baz{\b:baz}    % hook at end
\NewConfigure{baz}{2}       % declare \baz has two configurable hooks
```

Note that by default a `\NewConfigure{baz}{2}` command expects the hook corresponding to the first argument to be `\a:baz` and the hook corresponding to the second argument to be `\b:baz`. Extending the example, `\NewConfigure{foo}{9}` will expect the hook corresponding to the ninth argument to be `\i:foo` ('i' is the ninth letter of the alphabet). This default setting for `\NewConfigure` has been created via:
`\Configure{NewConfigure}{a:}{b:}{c:}{d:}{e:}{f:}{g:}{h:}{i:}`
in `tex4ht.sty`.

As an example for hook insertion, assume a macro defined like:
`\newcommand{\mac}[1]{START #1 END}`
in which there are four potential places for hooks (call them h1 to h4):
`{h1 START h2 #1 h3 END h4}`. Hooks h1 and h4 can be added via `\pend:defI` and `\append:defI`, but these are not sufficient by themselves. Other methods are required for inserting all four hooks. Two of these are:

- Redefine the whole macro from scratch:

  ```
  \renewcommand{\mac}[1]{\a:mac START \c:mac #1 \d:mac END \b:mac}
  \NewConfigure{mac}{4}
  ```

- Reuse parts of the original macro (similar to the LaTeX ap/pre-pending idiom):

```
\let\oldmac\mac
\renewcommand{\mac}[1]{\a:mac\oldmac{\c:mac #1 \d:mac}\b:mac}
\NewConfigure{mac}{4}
```

Either of these examples can be configured via:

```
\Configure{mac}%
  {first arg for a hook}    % \a:mac at the start of the command
  {second arg for a hook}   % \b:mac at the end of the command
  {third arg for a hook}    % \c:mac immediately before the argument
  {fourth arg for a hook}   % \d:mac immediately after the argument
```

Note that the hooks do not have to be placed in the `\mac` command in alphabetical order.

References

[GMS94] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*. Addison-Wesley Publishing Company, 1994.

[GR99] Michel Goossens and Sebastian Rahtz (with Eitan Gurari, Ross Moore, and Robert Sutor). *The LaTeX Web Companion — Integrating TeX, HTML, and XML*. Addison-Wesley Publishing Company, 1999.

[Wil96] Peter R. Wilson. *LaTeX for standards: The LaTeX package files user manual*. NIST Report NISTIR, June 1996.

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

U

V

W