POD Translation by *pod2pdf*

ajf@afco.demon.co.uk

html2wml.cgi

Table of Contents
html2wml.cgi

NAME

Html2Wml - Program that can convert HTML pages to WML pages

SYNOPSIS

in a shell:

```
html2wml.cgi [options] <file|url>
```

as a CGI:

```
/cgi-bin/html2wml.cgi?url=<url>
```

DESCRIPTION

Html2Wml converts HTML pages to WML pages, suitable for being viewed on a Wap device. The conversion can be done either on the command line to create static WML pages or on-the-fly by calling this program as a CGI.

As of version 0.3, the resulting WML should be well-formed, and in most cases valid. This is not guarantied but it should work for most HTML pages. To be more precise, the validity of the WML depends on the quality of the input HTML. Pages created with softwares that conform to W3C standards are most likely to produce valid WML. To check your HTML pages, your can use W3C's excellent software *HTML Tidy*, written by Dave Raggett.

OPTIONS

Note that most of these options can be used when calling Html2Wml as a CGI. See the file ***form.html*** in the ***t/*** directory for an example.

Conversion Options

—ascii

When this option is on, named HTML entities are converted to US-ASCII using the same 7 bit approximations as Lynx. By default, this is off, so that named entities are converted into numeric entities.

—collapse, —nocollapse

This option tells Html2Wml to collapse redundant white space characters and empty paragraphs. This option is on by default, but you can desactivate this by using —nocollapse.

This behavior is not really standard, but the aim is to reduce the size of the output. WML pages are primarily intented for Wap devices, which usually have slow connections. The smaller the WML result is, the faster it can be downloaded. Furthermore, collapsing white spaces is the normal behavior for HTML pages.

Empty paragraphs are also collapsed (this is really not standard), but it should avoid empty screens: the display of a Wap device is usually small, and it can be annoying to scroll down a lot because of many empty lines.

—compile

This option uses the WML compiler from WML Tools to convert the WML to a compact binary representation of the WML deck.

—hreftmpl=TEMPLATE

This options sets the template that will be used to reconstruct the `href` links.

See *"Links Reconstruction"* for more information.

—linearize, —nolinearize

This options is on by default. It makes Html2Wml flattens the tables *à la* Lynx. I think it is better than trying to use WML tables because, contrary to HTML tables, they have extremely limited features (in particular, they can't be nested). Therefore it's quite difficult to decide what to do when you have three nested tables. Furthermore, calculations on tables are quite CPU consuming, and Wap devices are not supposed to be powerful.

—nopre

This option tells Html2Wml not to use the `<pre>` tag. This is useful if you want to use the WML compiler from WML Tools 0.0.4, which doesn't recognize this tag.

—srctmpl=TEMPLATE

This options sets the template that will be used to reconstruct the `src` links.

See *"Links Reconstruction"* for more information.

Deck Splitting Options

—max-card-size=SIZE

This option allows you to limit the size of the generated cards. The value is given in bytes. Default is 1,500 bytes, which should be small enought to be loaded on any Wap device.

—card-split-threshold=SIZE

Splitting can occur when the size of the current card is between `max-card-size - card-split-threshold` and `max-card-size`.

—next-card-label=LABEL

This option sets the label of the link that allows the user to go to the next card. Default is "[>>]" (which will be rendered as "[]").

Debugging Options

—debug

This option activates the debug mode. This prints the output result with line numbering and with the result of the XML check. If the WML compiler was called, the result is also printed in hexadecimal an ascii forms. When called as a CGI, all of this is printed as HTML, so that can use any web browser for that purpose.

—xmlcheck

When this option is on, it send the WML output to XML::Parser to check its well-formedness.

FEATURES

Deck Splitting

In order to match the low memory capabilities of many Wap devices, Html2Wml allows you to convert the HTML document as a WML deck that contains several cards. The upper limit size of these cards can be set using the `max-card-size` option. This is not a guaranty as the size is calculated in an approximated way (if you wonder why I don‘t do an exact calculation, it‘s because it would be difficult in the current architecture of Html2Wml).

Actions

Actions are a feature similar to the SSI (Server Side Includes) available on web servers like Apache. In order not to interfere with real SSI, but to keep their syntax easy to learn, it differs in very few points.

Syntax

The syntax to execute an action is:

```
<!-- [action param1="value" param2='value'] -->
```

Note that the angle brackets are part of the syntax. Except for that point, Actions syntax is very similar to SSI syntax.

Available actions

include

Description

Includes a file in the document at the current point. Please note that Html2Wml doesn‘t check nor parse the file, and if the file cannot be found, will silently die (this is the same behavior as SSI).

Parameters

virtual=*url* — The file is get by http.

file=*path* — The file is read from the local disk.

Note

If you use the `file` parameter, an absolute path is recommend.

fsize

Description

Returns the size of a file at the current point of the document.

Parameters

You can use the same parameters as for the `include` action.

Examples

To include a small navigation bar:

```
<!-- [include virtual="nav.wml"] -->
```

Links Reconstruction

This engine allows you to reconstruct the links of the HTML document being converted. It has two modes, depending upon whether Html2Wml was launched from the shell or as a CGI.

When used as a CGI, this engine will reconstructs the links of the HTML document so that all the urls will be passed to Html2Wml in order to convert the pointed files (pages or images). This is completly automatic and can't be customized for now (but I don't think it would be really useful).

When used from the shell, this engine reconstructs the links with the URL template (the parameter of the `hreftmpl` option). Note that absolute URLs will be left untouched. The template can be customized using the following syntax. If no template is supplied, the links will be left untouched.

Syntax

The template is a string that contains the new URL. More precisely, it's a Text::Template template. Parameters can be interpolated as a constant or as a variable. The template is embraced between curcly bracets, and can contain any avlid Perl code.

The simplest form of a template is `{`*`PARAM`*`}` which just returns the value of *PARAM*. If you want to do something more complex, you can use the corresponding variable; for example `{"foo $`*`PARAM`*` bar"}`, or `{join '_', split ' ',` *`PARAM`*`}`.

You may read *Text::Template* for more information on what is possible within a template.

If the original URL contained a query part or a fragment part, then they will be appended to the result of the template.

Available parameters

`URL`

This parameter contains the original URL from the `href` or `src` attribute.

`FILENAME`

This parameter contains the base name of the file.

`FILEPATH`

This parameter contains the leading path of the file.

`FILETYPE`

This parameter contains the suffix of the file.

Examples

To add a path option:

```
{URL}$wap
```

Using Apache, you can then add a Rewrite directive so that URL ending with `$wap` will be redirected to Html2Wml:

```
RewriteRule  ^(/.*)\$wap$  /cgi-bin/html2wml.cgi?url=$1
```

To change the extension of an image:

```
{FILEPATH}{FILENAME}.wbmp
```

Note that `FILETYPE` contains all the extensions of the file, so its name is ***index.html.fr*** for example, `FILETYPE` contains "`.html.fr`".

CAVEATS

Currently, only the well-formedness of the resulting WML can be tested, not its validity.

Inverted tags (like "bold italic") may produce unexpected results. But only bad softwares do bad stuff like this.

LINKS

Html2Wml

http://www.resus.univ-mrs.fr/~madingue/techie/html2wml.html

HTML Tidy

http://www.w3.org/People/Raggett/tidy

WML Tools

http://pwot.co.uk/wml/

WAP Forum

http://www.wapforum.org/

You can find more links related to WAP and WML in the documentation of Html2Wml.

ACKNOWLEDGEMENTS

Werner Heuser - for his numerous ideas, advices and his help for the debugging

AUTHOR

Sébastien Aperghis-Tramoni <madingue@resus.univ-mrs.fr>

COPYRIGHT