

# wkhtmltopdf 0.9.6 Manual

This file documents wkhtmltopdf, a program capable of converting html documents into PDF documents.

## Contact

If you experience bugs or want to request new features please visit <http://code.google.com/p/wkhtmltopdf/issues/list>, if you have any problems or comments please feel free to contact me: see <http://www.madalgo.au.dk/~jakobt/#about>

## Reduced Functionality

Some versions of wkhtmltopdf are compiled against a version of QT without the wkhtmltopdf patches. These versions are missing some features, you can find out if your version of wkhtmltopdf is one of these by running `wkhtmltopdf --version` if your version is against an unpatched QT, you can use the static version to get all functionality.

Currently the list of features only supported with patch QT includes:

- Printing more then one HTML document into a PDF file.
- Running without an X11 server.
- Adding a document outline to the PDF file.
- Adding headers and footers to the PDF file.
- Generating a table of contents.
- Adding links in the generated PDF file.
- Printing using the screen media-type.
- Disabling the smart shrink feature of webkit.

## License

Copyright (C) 2008,2009 Wkhtmltopdf Authors.

License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>. This is free software: you are free to change and redistribute it. There is NO WARRANTY, to the extent permitted by law.

## Authors

Written by Jakob Truelsen. Patches by Mário Silva, Benoit Garret and Emmanuel Bouthenot.

## Synopsis

`wkhtmltopdf [OPTIONS]... <input file> [More input files] <output file>`

## General Options

	<code>--allow</code>	<code>&lt;path&gt;</code>	Allow the file or files from the specified folder to be loaded (repeatable)
<b>-b,</b>	<code>--book*</code>		Set the options one would usually set when printing a book
	<code>--collate</code>		Collate when printing multiple copies
	<code>--cookie</code>	<code>&lt;name&gt; &lt;value&gt;</code>	Set an additional cookie (repeatable)
	<code>--cookie-jar</code>	<code>&lt;path&gt;</code>	Read and write cookies from and to the supplied cookie jar file
	<code>--copies</code>	<code>&lt;number&gt;</code>	Number of copies to print into the pdf file (default 1)
	<code>--cover*</code>	<code>&lt;url&gt;</code>	Use html document as cover. It will be inserted before the toc with no headers and footers
	<code>--custom-header</code>	<code>&lt;name&gt; &lt;value&gt;</code>	Set an additional HTTP header (repeatable)
	<code>--debug-javascript</code>		Show javascript debugging output
<b>-H,</b>	<code>--default-header*</code>		Add a default header, with the name of the page to the left, and the page number to the right, this is short for: <code>--header-left='[webpage]' --header-right='[page]/[toPage]' --top 2cm --header-line</code>
	<code>--disable-external-links*</code>		Do not make links to remote web pages
	<code>--disable-internal-links*</code>		Do not make local links
<b>-n,</b>	<code>--disable-javascript</code>		Do not allow web pages to run javascript
	<code>--disable-pdf-compression*</code>		Do not use lossless compression on pdf objects
	<code>--disable-smart-shrinking*</code>		Disable the intelligent shrinking strategy used by WebKit that makes the pixel/dpi ratio none constant
	<code>--disallow-local-file-access</code>		Do not allowed conversion of a local file to read in other local files, unless explicity allowed with <code>--allow</code>
<b>-d,</b>	<code>--dpi</code>	<code>&lt;dpi&gt;</code>	Change the dpi explicitly (this has no effect on X11 based systems)
	<code>--enable-plugins</code>		Enable installed plugins (such as flash)
	<code>--encoding</code>	<code>&lt;encoding&gt;</code>	Set the default text encoding, for input
	<code>--extended-help</code>		Display more extensive help, detailing less common command switches
	<code>--forms*</code>		Turn HTML form fields into pdf form fields
<b>-g,</b>	<code>--grayscale</code>		PDF will be generated in grayscale
<b>-h,</b>	<code>--help</code>		Display help
	<code>--htmldoc</code>		Output program html help
	<code>--ignore-load-errors</code>		Ignore pages that clames to have encountered an error during loading
<b>-l,</b>	<code>--lowquality</code>		Generates lower quality pdf/ps. Useful to shrink the result document space
	<code>--manpage</code>		Output program man page
<b>-B,</b>	<code>--margin-bottom</code>	<code>&lt;unitreal&gt;</code>	Set the page bottom margin (default 10mm)

-L, --margin-left	<unitreal>	Set the page left margin (default 10mm)
-R, --margin-right	<unitreal>	Set the page right margin (default 10mm)
-T, --margin-top	<unitreal>	Set the page top margin (default 10mm)
--minimum-font-size	<int>	Minimum font size (default 5)
--no-background		Do not print background
-O, --orientation	<orientation>	Set orientation to Landscape or Portrait
--page-height	<unitreal>	Page height (default unit millimeter)
--page-offset*	<offset>	Set the starting page number (default 1)
-s, --page-size	<size>	Set paper size to: A4, Letter, etc.
--page-width	<unitreal>	Page width (default unit millimeter)
--password	<password>	HTTP Authentication password
--post	<name> <value>	Add an additional post field (repeatable)
--post-file	<name> <path>	Post an additional file (repeatable)
--print-media-type*		Use print media-type instead of screen
-p, --proxy	<proxy>	Use a proxy
-q, --quiet		Be less verbose
--read-args-from-stdin		Read command line arguments from stdin
--readme		Output program readme
--redirect-delay	<msec>	Wait some milliseconds for js-redirects (default 200)
--replace*	<name> <value>	Replace [name] with value in header and footer (repeatable)
--stop-slow-scripts		Stop slow running javascripts
--title	<text>	The title of the generated pdf file (The title of the first document is used if not specified)
-t, --toc*		Insert a table of content in the beginning of the document
--use-xserver*		Use the X server (some plugins and other stuff might not work without X11)
--user-style-sheet	<url>	Specify a user style sheet, to load with every page
--username	<username>	HTTP Authentication username
-V, --version		Output version information an exit
--zoom	<float>	Use this zoom factor (default 1)

Items marked \* are only available using patched QT.

## Headers And Footer Options

--footer-center*	<text>	Centered footer text
--footer-font-name*	<name>	Set footer font name (default Arial)
--footer-font-size*	<size>	Set footer font size (default 11)
--footer-html*	<url>	Adds a html footer
--footer-left*	<text>	Left aligned footer text
--footer-line*		Display line above the footer
--footer-right*	<text>	Right aligned footer text
--footer-spacing*	<real>	Spacing between footer and content in mm (default 0)
--header-center*	<text>	Centered header text
--header-font-name*	<name>	Set header font name (default Arial)
--header-font-size*	<size>	Set header font size (default 11)
--header-html*	<url>	Adds a html header
--header-left*	<text>	Left aligned header text
--header-line*		Display line below the header
--header-right*	<text>	Right aligned header text
--header-spacing*	<real>	Spacing between header and content in mm (default 0)

Items marked \* are only available using patched QT.

## Table Of Content Options

--toc-depth*	<level>	Set the depth of the toc (default 3)
--toc-disable-back-links*		Do not link from section header to toc
--toc-disable-links*		Do not link from toc to sections
--toc-font-name*	<name>	Set the font used for the toc (default Arial)
--toc-header-font-name*	<name>	The font of the toc header (if unset use --toc-font-name)
--toc-header-font-size*	<size>	The font size of the toc header (default 15)
--toc-header-text*	<text>	The header text of the toc (default Table Of Contents)
--toc-l1-font-size*	<size>	Set the font size on level 1 of the toc (default 12)
--toc-l1-indentation*	<num>	Set indentation on level 1 of the toc (default 0)
--toc-l2-font-size*	<size>	Set the font size on level 2 of the toc (default 10)
--toc-l2-indentation*	<num>	Set indentation on level 2 of the toc (default 20)
--toc-l3-font-size*	<size>	Set the font size on level 3 of the toc (default 8)
--toc-l3-indentation*	<num>	Set indentation on level 3 of the toc (default 40)
--toc-l4-font-size*	<size>	Set the font size on level 4 of the toc (default 6)
--toc-l4-indentation*	<num>	Set indentation on level 4 of the toc (default 60)
--toc-l5-font-size*	<size>	Set the font size on level 5 of the toc (default 4)
--toc-l5-indentation*	<num>	Set indentation on level 5 of the toc (default 80)
--toc-l6-font-size*	<size>	Set the font size on level 6 of the toc (default 2)
--toc-l6-indentation*	<num>	Set indentation on level 6 of the toc (default 100)
--toc-l7-font-size*	<size>	Set the font size on level 7 of the toc (default 0)

<b>--toc-l7-indentation*</b>	<code>&lt;num&gt;</code>	Set indentation on level 7 of the toc (default 120)
<b>--toc-no-dots*</b>		Do not use dots, in the toc

Items marked \* are only available using patched QT.

## Outline Options

<b>--dump-outline*</b>	<code>&lt;file&gt;</code>	Dump the outline to a file
<b>--outline*</b>		Put an outline into the pdf
<b>--outline-depth*</b>	<code>&lt;level&gt;</code>	Set the depth of the outline (default 4)

Items marked \* are only available using patched QT.

## Specifying A Proxy

By default proxy information will be read from the environment variables: proxy, all\_proxy and http\_proxy, proxy options can also be specified with the -p switch

```
<type> := "http://" | "socks5://"
<serif> := <username> (":" <password>)? "@"
<proxy> := "None" | <type>? <serif>? <host> (":" <port>)?
```

Here are some examples (In case you are unfamiliar with the BNF):

```
http://user:password@myproxyserver:8080
socks5://myproxyserver
None
```

## Footers And Headers

Headers and footers can be added to the document by the --header-\* and --footer\* arguments respectfully. In header and footer text string supplied to e.g. --header-left, the following variables will be substituted.

* [page]	Replaced by the number of the pages currently being printed
* [frompage]	Replaced by the number of the first page to be printed
* [topage]	Replaced by the number of the last page to be printed
* [webpage]	Replaced by the URL of the page being printed
* [section]	Replaced by the name of the current section
* [subsection]	Replaced by the name of the current subsection
* [date]	Replaced by the current date in system local format
* [time]	Replaced by the current time in system local format

As an example specifying --header-right "Page [page] of [toPage]", will result in the text "Page x of y" where x is the number of the current page and y is the number of the last page, to appear in the upper left corner in the document.

Headers and footers can also be supplied with HTML documents. As an example one could specify --header-html header.html, and use the following content in header.html:

```
<html><head><script>
function subst() {
  var vars={};
  var x=document.location.search.substring(1).split('&');
  for(var i in x) {var z=x[i].split('=');vars[z[0]] = unescape(z[1]);}
  var x=['frompage','topage','page','webpage','section','subsection','subsubsection'];
  for(var i in x) {
    var y = document.getElementsByClassName(x[i]);
    for(var j=0; j<y.length; ++j) y[j].textContent = vars[x[i]];
  }
}
</script></head><body style="border:0; margin: 0;" onload="subst()">
<table style="border-bottom: 1px solid black; width: 100%">
  <tr>
    <td class="section"></td>
    <td style="text-align:right">
      Page <span class="page"></span> of <span class="topage"></span>
    </td>
  </tr>
</table>
</body></html>
```

As can be seen from the example, the arguments are sent to the header/footer html documents in get fashion.

## Outlines

Wkhtmltopdf with patched qt has support for PDF outlines also known as book marks, this can be enabled by specifying the --outline switch. The outlines are generated based on the <h?> tags, for a in-depth description of how this is done see the "Table Of Contest" section.

The outline tree can sometimes be very deep, if the <h?> tags were spread to generous in the HTML document. The --outline-depth switch can be used to bound this.

## Page Breaking

The current page breaking algorithm of WebKit leaves much to be desired. Basically webkit will render everything into one long page, and then cut it up into pages. This means that if you have two columns of text where one is vertically shifted by half a line. Then webkit will cut a line into to pieces display the top half on one page. And the bottom half on another page. It will also break image in two and so on. If you are using the patched version of QT you can use the CSS page-break-inside property to remedy this somewhat. There is no easy solution to this problem, until this is solved try organising your HTML documents such that it contains many lines on which pages can be cut cleanly.

See also: <http://code.google.com/p/wkhtmltopdf/issues/detail?id=9>, <http://code.google.com/p/wkhtmltopdf/issues/detail?id=33> and

<http://code.google.com/p/wkhtmltopdf/issues/detail?id=57>.

## Page sizes

The default page size of the rendered document is A4, but using this `--page-size` option this can be changed to almost anything else, such as: A3, Letter and Legal. For a full list of supported pages sizes please see <http://doc.trolltech.com/4.6/qprinter.html#PageSize-enum>.

For a more fine grained control over the page size the `--page-height` and `--page-width` options may be used

## Reading arguments from stdin

If you need to convert a lot of pages in a batch, and you feel that wkhtmltopdf is a bit to slow to start up, then you should try `--read-args-from-stdin`,

When `--read-args-from-stdin` each line of input sent to wkhtmltopdf on stdin will act as a separate invocation of wkhtmltopdf, with the arguments specified on the given line combined with the arguments given to wkhtmltopdf

For example one could do the following:

```
echo "http://doc.trolltech.com/4.5/qapplication.html qapplication.pdf" >> cmds
echo "--cover google.com http://en.wikipedia.org/wiki/Qt_(toolkit) qt.pdf" >> cmds
wkhtmltopdf --read-args-from-stdin --book < cmds
```

## Static version

On the wkhtmltopdf website you can download a static version of wkhtmltopdf <http://code.google.com/p/wkhtmltopdf/downloads/list>. This static binary will work on most systems and comes with a build in patched QT.

Unfortunately the static binary is not particularly static, on Linux it depends on both glibc and openssl, furthermore you will need to have an xserver installed but not necessary running. You will need to have different fonts install including xfonts-scalable (Type1), and msttcorefonts. See <http://code.google.com/p/wkhtmltopdf/wiki/static> for trouble shouting.

## Compilation

It can happen that the static binary does not work for your system for one reason or the other, in that case you might need to compile wkhtmltopdf yourself.

### GNU/Linux:

Before compilation you will need to install dependencies: X11, gcc, git and openssl. On Debian/Ubuntu this can be done as follows:

```
sudo apt-get build-dep libqt4-gui libqt4-network libqt4-webkit
sudo apt-get install openssl build-essential xorg git-core git-doc libssl-dev
```

On other systems you must use your own package manager, the packages might be named differently.

First you must check out the modified version of QT

```
git clone git://gitorious.org/+wkhtml2pdf/qt/wkhtmltopdf-qt.git wkhtmltopdf-qt
```

Next you must configure, compile and install QT, note this will take quite some time, depending on what arguments you use to configure qt

```
cd wkhtmltopdf-qt
./configure -nomake tools,examples,demos,docs,translations -opensource -prefix ../wkqt
make -j3
make install
cd ..
```

All that is needed now is, to compile wkhtmltopdf.

```
git clone git://github.com/antialize/wkhtmltopdf.git wkhtmltopdf
cd wkhtmltopdf
../wkqt/bin/qmake
make -j3
```

You should now have a binary called wkhtmltopdf in the currently folder that you can use, you can optionally install it by running

```
make install
```

### Other operative systems and advanced features

If you want more details or want to compile under other operative systems other than GNU/Linux, please see <http://code.google.com/p/wkhtmltopdf/wiki/compilation>.

## Installation

There are several ways to install wkhtmltopdf. You can download a already compiled binary, or you can compile wkhtmltopdf yourself. On windows the easiest way to install wkhtmltopdf is to download the latest installer. On linux you can download the latest static binary, however you still need to install some other pieces of software, to learn more about this read the static version section of the manual.

## Examples

This section presents a number of examples of how to invoke wkhtmltopdf.

To convert a remote HTML file to PDF:

```
wkhtmltopdf http://www.google.com google.pdf
```

To convert a local HTML file to PDF:

```
wkhtmltopdf my.html my.pdf
```

You can also convert to PS files if you like:

```
wkhtmltopdf my.html my.ps
```

Produce the eler2.pdf sample file:

```
wkhtmltopdf http://geekz.co.uk/lovesraymond/archive/eler-highlights-2008 eler2.pdf -H --outline
```

**Generated : Fri Jul 12 17:40:18 BST 2013**

<http://madalgo.au.dk/~jakobt/wkhtmltoxdoc/wkhtmltopdf-0.9.9-doc.html>