

PDF

Contents

1. [PDF](#)
 1. [Background and rationale](#)
 2. [Description](#)
 3. [Usage](#)
 4. [Prerequisites](#)
 5. [Download & Release Notes](#)
 6. [Examples](#)
 7. [History](#)
 8. [Copyright](#)
 9. [License](#)
 10. [Known issues and limitations](#)

Appendices

1. [Settings](#)

Background and rationale

[MoinMoin](#) pages can be converted to a PDF in several ways. There are existing plugin actions at <http://moinmo.in/ActionMarket>. Also, a PDF can be made by saving the print view from a browser as a PDF (under OS X) or using a PDF print driver in another system. Under OS X, creating a PDF from Safari works moderately well, but not so for Firefox. The main constraint is that links within the page (e.g. as generated by TableOfContents macro) are directed back to the page URL with #target appended. So the PDF is not independent and is potentially annoying, especially if the source page is not publicly readable. This primary PDF could be printed again to create a secondary PDF, which removes these links. However, this is not a solution, it is just an avoidance mechanism.

The two plugin actions on offer, as of Feb 2013, at the [ActionMarket](#) have their merits, though these seem to be less than simple to install and have other constraints.

A method based on [wkhtmltopdf](#) would be relatively simple and provides a PDF of workable quality for most [MoinMoin](#) pages.

Description

The PDF action allows the creation of a PDF version of a [MoinMoin](#) page using [wkhtmltopdf](#), a "simple shell utility to convert html to pdf using the WebKit rendering engine and qt."

As implemented here, the PDF created has headers and footers with page title, date-time, url and page numbers. Also the page is set to A4 with 20 mm margins. If other defaults are preferred, the code can be edited accordingly. Also, some settings can be overridden using the #pragma processing instruction on a page by page basis.

The PDF created by this action is stored in the page's cache, and served from there if not needing to be refreshed (e.g after a page has been edited). Selecting Delete Cache from the More Actions: menu or adding the query string ?action=refresh to the URL will delete the cached PDF. It will be recreated the next time the PDF action is invoked.

Usage

The action can be called from the More actions: list or added to the page as <<Action(PDF)>>.


However in the latter case, the link to the action will also appear in the generated PDF.

The addition of the #action processing instruction (see [PiAction](#)) allows for the PDF action to be set as the default for the page (i.e. rather than action=show, the current default). So by including #action PDF at the start of a page, the PDF version of the page will be the default view.

The action can also be evoked by appending the page URL with ?action=PDF, and if you are the superuser, you can append ?action=PDF&show=call to show the call created but not sent to the subprocess. This can be cut and pasted into your terminal and run locally, if desired. The reason for this is that the behaviour of wkhtmltopdf seems to be system dependent, so you might get a more suitable result from your system rather than from the wiki's server, assuming you have wkhtmltopdf installed. The call sets --cookie for wkhtmltopdf, allowing you to access the [MoinMoin](#) page as a logged in user.

In this use of wkhtmltopdf, the action sets some settings differently from those of wkhtmltopdf. It is possible to override a selected number of wkhtmltopdf settings using processing instructions #pragma pdf <setting> <value>. For details of this, see the appendix [Settings](#).

Prerequisites

To allow PDF action to work  [wkhtmltopdf](#) needs to be installed. Also, the variable pdf_app has to be added to wikiconfig.py and set to the local path for wkhtmltopdf.

To allow for refresh action to delete the cached pdf, refresh.py needs to be copied to wiki/data/plugin/action folder and the following line inserted as the second last line in the code (a modified copy of refresh.py is included in the Download section).

Toggle line numbers







```
1 ...
2 caching.CacheEntry(request, arena, "text_pdf.pdf", scope="item").remove()
3 request.page.send_page()
```

If PDFs are also created for slides (see [SlideSet.py](#)) then an additional cached file needs to be removed by the refresh action. To achieve this, add the following code to the local copy of refresh.py as above.

Toggle line numbers


```
1 caching.CacheEntry(request, arena, "slides_pdf.pdf", scope="item").remove()
```


Download & Release Notes

Download	Release Version	Moin Version	Release Notes
 PDF-1.2.py	1.2	1.9.2	
 PDF-1.3.py	1.3	1.9.2	Modified to allow call from GuestPass action
 PDF-1.4.py	1.4	1.9.2	Caching for pdf and wkhtmltopdf path in cfg
 PDF-1.5.py	1.5	1.9.2	Enabled SlideSet pdf and minor fixes
 PDF-1.6.py	1.6	1.9.8	Unquoted URLs for display
 refresh.py		1.9.2	Modified for pdf caching


For installation, rename files to remove version number (e.g. -1.0).

Examples

See  [PDF.py action.pdf](#), for this page rendered as PDF by this action.

Compare it with  [PDF.py safari.pdf](#), for this page saved as a PDF from Safari.





It is worth noting the difference in how links are handled and the inclusion of a PDF outline in the version created by this action. Also, margin width cannot be controlled when saving a PDF from Safari, and being narrower, the page layout is less visually appealing.

Alternatively, try it directly from the Action macro link:  [PDF](#).

History

See individual source files for details of version history.

Developed with inspiration and [MoinMoin](#) code from:

@copyright: 2000-2004 Juergen Hermann <  jh@web.de >
 @copyright: 2000-2001 Richard Jones <  richard@bizarsoftware.com.au >
 @copyright: 2001 Ken Sugino ( sugino@mediaone.net)
 @copyright: 2005-2010 [MoinMoin: ReimarBauer](#)
 @copyright: 2005 [MoinMoin: AlexanderSchremmer](#)
 @copyright: 2005 Diego Ongaro at ETSZONE ( diego@etszone.com)
 @copyright: 2006-2008 [MoinMoin: ThomasWaldmann](#),



Copyright

@copyright: 2013, 2014 Ian Riley <  ian@riley.asia >

License

GNU GPL, see COPYING for details.

Known issues and limitations

- wkhtmltopdf 0.10.0 rc2 running under OS X constructs correct links to other pages from the same domain, but wkhtmltopdf 0.11.0 rc1 under GNU/Linux (this site) does not. For example, in the latter case a link <http://rileylink.net/moin.cgi/PDF.py> is constructed as the non-functional </moin.cgi/PDF.py>. See the  [PDF.py action.pdf](#) verses  [PDF.py action osx.pdf](#) and look at the links to these two files.
 Although this remains as an issue, it can be avoided by using the [collect.py](#) parser, which processes the page after it is rendered in HTML to correct linking before sending it to wkhtmltopdf for creation of the PDF. Although the [collect.py](#) parser was written to combined several pages, it can be used for a single page if functional internal linking is required.
- 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." [from [wkhtmltopdf](#) documentation]

- ~~The output file created by wkhtmltopdf is currently placed under wiki/data, so needs to be recreated every time the action is called. This reduces performance, and it might be better to place the file under wiki/data/pages/[pagename]/cache, and only recreate it if the cached version is older than the latest version of the page. This option would improve performance but, of course, increase storage requirements. It was not done initially because MoinMoin installed under the OS X WebServer (the development platform) doesn't provide read/write access to the wkhtmltopdf subprocess for that latter location. This could be solved by having the MoinMoin action decide if the file needs to be recreated and to move it to the storage location when updated.~~

Caching introduced with version 1.4.

- The cached version of the PDF might contain non-current dates or perhaps user-specific information, such as inserted by [UserInfo.py](#) or similar macros. In such circumstances, it would be advisable to delete the cache if current dates are needed, or to remove copies with user-specific information after saving the PDF. Setting the [PiAction](#) processing instruction to `#action refresh` will force cache to be cleared before showing the page, which useful for some pages, however, it cannot also be used to force the page to be delivered as a PDF.

Hits

3627

PDF.py (last edited 2017-09-09 04:49:50 by IanRiley)