EXT: Webkit PDF

Extension Key: webkitpdf

Language: en

Table of Contents

Introduction

What does it do?

Webkitpdf renders given pages to a PDF file on the fly. The rendering is done by the Webkit engine, which is shipped as a binary script with the extension.

Once rendered, the PDF files are cached in the filesystem for a better performance.

The caching information is kept in a database table. Clearing the cache in the TYPO3 backend will remove those PDF files which exceed a certain lifetime limit. This limit can be set in the configuration section of the extension manager.

IMPORTANT NOTICES

- A binary script (res/wkhtmltopdf) is included to perform this task. The user must have the right permissions to execute this binary (script has to be executable). You can move it to another location and specify the new path via TypoScript.
- This extension does NOT work on Windows systems!
- The binary provided by this extension does only work on 32bit systems. To run this extension on a 64bit system, download the binary provided on forge.typo3.org/issues/show/4337 and use the TypoScript setting customScriptPath to point to the downloaded binary.
- The binary script apparently has some dependencies. Make sure you have libfreetype installed. It may occur that libfreetype alone isn't enough. In this case we recommend to install GLIBC > 2.4.
- If you experience problems (e.g. the PDF is not readable), try to run the script manually from a shell environment. To get the exact call which was made by WebkitPDF, use the debugScriptCall setting in TypoScript.

Upgrading from version 1.3.1 (or below) to any later version

Link generation has changed in versions >1.3.1 ! You need to perform some manual steps to update your environment.

The userfunc is not needed anymore and its use is deprecated. Instead, you have to use a built-in TypoScript object plugin.tx_webkitpdf_pi1.pdfLink

Follow these steps after upgrading from 1.3.1 (or below) to any later version:

The old userfunc TypoScript which is going to be replaced should look like this:

```
includeLibs.webkit = EXT:webkitpdf/res/user_webkitpdf.php
lib.pdf = USER
lib.pdf.userFunc = user_webkitpdf->user_getPDFLink
lib.pdf {
  pid = 23
  linkText = Save as PDF
}
```

- First, go to your TypoScript template and include the static template from extension 'WebKit PDF' in your template.
- Go to template constant editor. Set pluginPid according to lib.pdf.pid (it's the page id which contains the plugin 'WebKit PDF'). The text of the link can be set in linkText.
- Remove the old TypoScript as shown above and replace it by the below line:

```
lib.pdf < plugin.tx_webkitpdf_pi1.pdfLink
```

Administration

- Install via extension manager.
- Create a new page and add a plugin 'Webkit PDF'.
- Go to your TypoScript template and include the static template from extension 'WebKit PDF' in your template.
- Go to template constant editor. Set pluginPid according to the page id with the plugin (of step 2). The text of the link can be changed in linkText.
- Integrate the link for PDF generation in your page using the below TypoScript example.
- Pass the page IDs to render in GET-parameter 'urls' (or custom if specified via TS)
- Parameter is an array of URLs

Example:

```
page.70 < plugin.tx_webkitpdf_pi1.pdfLink
```

See the configuration section of plugin.tx_webkitpdf_pi1.pdfLink for more details on how the link is generated.

Configuration

Reference

General Options

Property:	Data type:	Description:	Default:
filePrefix	String/stdWrap	Custom prefix for the name of the generated PDF file	
customScriptPath	String /stdWrap	If you copy the used shell script to another location for permission reasons, you can enter the new path to the executable.	
customTempOutputPath	String/stdWrap	Custom path to store the temp files	typo3temp
copyrightNotice	String/stdWrap	Enter a name of a person or organisation to be shown as a copyright notice. Copyright 2009 [name]	
additionalStylesheet	String/stdWrap	Enter path to an additional CSS file.	
customParameterName	String/stdWrap	Custom name of the GET parameter containing the page IDs	selected_pages
pageURLInHeader	Boolean (1/0)/stdWrap	Add the URL of the page to PDF header	0
debugScriptCall	Boolean (1/0)/stdWrap	Prints the call of the script to the screen for debugging purposes. You can copy/paste this call into the shell to see if the call itself throws errors.	0
staticFileName	String/stdWrap	Enter a filename for the downloadable PDF. You can use a cObj to add dynamic things like timestamp to the filename. If you leave this option empty, the filename will be the same as the name of the temporary file.	

scriptParams	Array	Enter parameters for the script call directly.
		See section “Script parameters” for a full list.
		
		Enter parameters this way:
		
		scriptParams {
		    margin-left = 35mm
		    footer-line =
		  }
		
		The script execution will look this way:
		
		wkhtmltopdf --margin-left 35mm --footer-line ...	

Property:	Data type:	Description:	Default:

pdfLink	cObj	Pre-built cObject to link to PDF generation.
		
		pdfLink is a helper object to make the link generation as easy as possible. It's use is optional, you can create the link however you like. It uses the following TypoScript:
		
		setup.txt:
		`plugin.tx_webkitpdf_pi1.pdfLink = TEXT`
		`plugin.tx_webkitpdf_pi1.pdfLink {`
		`  value = {$plugin.tx_webkitpdf_pi1.pdfLink.linkText}`
		`  typolink {`
		`    parameter = {$plugin.tx_webkitpdf_pi1.pdfLink.pluginPid}`
		`    no_cache = {$plugin.tx_webkitpdf_pi1.pdfLink.no_cache}`
		`    additionalParams {`
		`      data = getIndpEnv:TYPO3_REQUEST_URL`
		`      wrap = &tx_webkitpdf_pi1[urls][0]=|`
		`    }`
		`  }`
		`}`
		
		constants.txt:
		`plugin.tx_webkitpdf_pi1.pdfLink {`
		` pluginPid = 123`
		` linkText = Save as PDF`
		` no_cache = 1`
		`}`
		
		The HTML result is:
		` <a href="http://www.example.com/index.php?id=123&no_cache=1&tx_webkitpdf_pi1[urls][0]=http://www.example.com/index.php?id=1">Save as PDF</a>`
		
		You simply have to use plugin.tx_webkitpdf_pi1.pdfLink in your page template.
		The pages which should be rendered as PDF have to be set as URL in tx_webkitpdf_pi1[urls][]. If you like to have one page rendered, use:
		&tx_webkitpdf_pi1[urls][0]=URL
		If you like to have multiple pages rendered in one PDF file, just add more parameters:
		&tx_webkitpdf_pi1[urls][1]=URL&tx_webkitpdf_pi1[urls][2]=URL and so on.
		Since cObject TEXT with typolink is used here, you have stdWrap which opens up infinite ways to realize your favorite link.
			cObj TEXT with typolink (see description)

[plugin.tx_webkitpdf_pi1]

Script parameters

This is a list of available parameters. When using this in TypoScript omit the '--'.

Property:	Description:	Default:
--book	Set the options one would usualy set when printing a book.	
--cover <url>	Use html document as cover. It will be inserted before the toc with no headers and footers.	
--default-header	Add a default header, with the name of the page to the left, and the page number to the right, this is short for: --header-left='[webpage]' –header-right='[page]/[toPage]' --top 2cm --header-line.	
--disable-javascript	Do not allow webpages to run javascript.	
--dpi <dpi>	Change the dpi explicitly.	
--enable-plugins	Enable installed plugins (such as flash). Note: This doesn't really work. The developer's hope to integrate it in one of the next versions.	
--encoding <encoding>	Set the default text encoding, for input.	
--footer-center <text>	Centered footer text.	

Property:	Description:	Default:
--footer-font-name <name>	Set footer font name.	Arial
--footer-font-size <size>	Set footer font size.	11
--footer-left <text>	Left aligned footer text.	
--footer-line	Display line above the footer.	
--footer-right <text>	Right aligned footer text.	
--grayscale	PDF will be generated in grayscale.	
--header-center <text>	Centered header text.	
--header-font-name <name>	Set header font name.	Arial
--header-font-size <size>	Set header font size.	11
--header-left <text>	Left aligned header text.	
--header-line	Display line below the header.	
--header-right <text>	Right aligned header text.	
--lowquality	Generates lower quality pdf/ps. Useful to shrink the result document space.	
--margin-bottom <unitread>	Set the page bottom margin.	10mm
--margin-left <unitread>	Set the page left margin.	10mm
--margin-right <unitread>	Set the page right margin.	10mm
--margin-top <unitread>	Set the page top margin.	10mm
--no-background	Do not print background.	
--orientation <orientation>	Set orientation to Landscape or Portrait.	
--outline	Put an outline into the pdf.	
--outline-depth <level>	Set the depth of the outline.	4
--page-offset <offset>	Set the starting page number.	1
--page-size <size>	Set pape size to: A4, Letter, ect..	
--password <password>	HTTP Authentication password.	
--print-media-type	Use print media-type instead of screen.	
--proxy <proxy>	Use a proxy.	
--redirect-delay <msec>	Wait some miliseconds for js-redirects,	200
--toc	Insert a table of content in the beginning of the document.	
--toc-depth <level>	Set the depth of the toc.	3
--toc-font-name <name>	Set the font used for the toc.	Arial
--toc-header-fs <size>	The font size of the toc header	15
--toc-header-text <text>	The header text of the toc.	Table Of Contents
--toc-l1-font-size <size>	Set the font size on level 1 of the toc.	12
--toc-l1-indentation <num>	Set indentation on level 1 of the toc.	0
--toc-l2-font-size <size>	Set the font size on level 2 of the toc.	10
--toc-l2-indentation <num>	Set indentation on level 2 of the toc.	20
--toc-l3-font-size <size>	Set the font size on level 3 of the toc.	8
--toc-l3-indentation <num>	Set indentation on level 3 of the toc.	40
--toc-no-dots	Do not use dots, in the toc.	
--user-style-sheet <url>	Specify a user style sheet, to load with every page.	
--username <username>	HTTP Authentication username.	

Note: The following parameters will override some TypoScript settings:

Parameter	Overwrites this TypoScript setting
user-style-sheet	additionalStylesheet
footer-left	copyrightNotice

header-center	pageURLInHeader

Additional script information

Proxy:

```
  By default proxy information will be read from the environment
  variables: proxy, all_proxy and http_proxy, proxy options can
  also by specified with the -p switch
  <type> := "http://" | "socks5://"
  <userinfo> := <username> (":" <password>)? "@"
  <proxy> := "None" | <type>? <userinfo>? <host> (":" <port>)?
```

Header and footer text:

In a header or footer text the following variables can be used

```
 * [page]       Replaced by the number of the pages currently beeing 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 beeing printed
 * [section]    Replaced by the name of the current section
 * [subsection] Replaced by the name of the current subsection
```

Known problems

- None currently

To-Do list

– Waiting for Flash support. This would be really awesome!

ChangeLog

- 1.0.0: initial release.
- 1.0.1:
  - Fixed usage with multiple GET parameters
  - Added userFunc to generate PDF link
  - New TS option: pageURLInHeader
  - Parameter linkText for userFunc can be any TS object now
- 1.1.0
  - Extension caches the rendered PDF files for reuse. Clear cache in backend.
  - Caching is done in a DB table. Clearing the cache clears temporary files older than 10 minutes.
- 1.1.1
  - Set any available script parameter via TypoScript. This gives you full flexibility.
- 1.1.2
  - TypoScript-Option to let WebkitPDF return only the name of the generated file.
- 1.1.3
  - Removed unnecessary code from ext_tables.php
  - Fixed issue with the PDF download occurring in some browsers
- 1.1.4
  - Security fix
- 1.1.5
  - Fixed caching
  - Cache files only if cache threshold is > 0
  - Every TS option (except scriptParams) has stdWrap now
  - Moved some methods to a new utils class