

EXT: Webkit PDF

Extension Key: webkitpdf

Language: en

Copyright 2009, Dev-Team Typoheads, <dev@typoheads.at>

This document is published under the Open Content License
available from <http://www.opencontent.org/opl.shtml>

The content of this document is related to TYPO3
- a GNU/GPL CMS/Framework available from www.typo3.org

Table of Contents

EXT: Webkit PDF.....	1	Troubleshooting.....	7
Introduction.....	3	Upgrading from version 1.1.3 (or below) to any later version.....	7
What does it do?.....	3	Generating PDFs of access protected pages.....	7
IMPORTANT NOTICES.....	3	Warnings on PDF generation page.....	7
Administration.....	4	Use a page type to render PDFs.....	8
Configuration.....	5	Known problems.....	9
Reference.....	5	To-Do list.....	10
General Options.....	5	ChangeLog.....	11
Script parameters.....	6		

Introduction

What does it do?

WebkitPDF uses the binary "wkhtmltopdf" (<http://code.google.com/p/wkhtmltopdf/>) to render given pages to a PDF file on the fly. The binary for 32bit i386 unix systems is shipped 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.

Administration

- Install via extension manager.
- Create a new page and add a plugin 'Webkit PDF'.
- Go to your TypeScript 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 TypeScript 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:
allowedHosts	String	Comma separated list of allowed hosts for PDF generation. WebkitPDF only allows the current host per 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 2010 [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 stdWrap 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.	
openFilesInline	Boolean (1/0)/stdWrap	Normally, the PDF files are sent with a "Content-Disposition" header "attachment" which causes the browser to show a "Save or open" dialog. If "openFilesInline" is set to "1", the PDF files are opened inline in the browser. If the browser doesn't know how to show a PDF file inline, the download dialog is shown.	0
disableCache	Boolean (1/0)/stdWrap	Overrides the settings made in extension manager. Use this to disable the cache based on certain conditions. If "disableCache" is set to "1", no files are stored in typo3temp/tx_webkitpdf	0
scriptParams	Array Every setting inside: String/stdWrap	Enter parameters for the script call directly. See section " Script parameters " for a full list. Enter parameters this way: <pre>scriptParams { margin-left = 35mm footer-line = }</pre> The script execution will look this way: wkhtmltopdf --margin-left 35mm --footer-line ...	

Property:	Data type:	Description:	Default:
pdfLink	cObj	<p>Pre-built cObject to link to PDF generation.</p> <p>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:</p> <p>setup.txt:</p> <pre>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]= } } }</pre> <p>constants.txt:</p> <pre>plugin.tx_webkitpdf_pi1.pdfLink { pluginPid = 123 linkText = Save as PDF no_cache = 1 }</pre> <p>The HTML result is:</p> <pre>Save as PDF</pre> <p>You simply have to use plugin.tx_webkitpdf_pi1.pdfLink in your page template.</p> <p>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</p> <p>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.</p> <p>Since cObject TEXT with typolink is used here, you have stdWrap which opens up infinite ways to realize your favorite link.</p>	cObj TEXT with typolink (see description)

[plugin.tx_webkitpdf_pi1]

Script parameters

A list of available parameters can be found here:

http://madalgo.au.dk/~jakobt/wkhtmltopdf-0.10.0_beta2-doc.html

When using this in TypoScript omit the '--'.

Note: The following parameters will override some TypoScript settings:

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

Troubleshooting

Upgrading from version 1.1.3 (or below) to any later version

Link generation has changed in versions >1.1.3 ! 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 1.1.3 (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
```

Generating PDFs of access protected pages

WebkitPDF is capable of generating PDFs of access protected pages. But in order to be able to do so, you have to do some configuration in the install tool:

- Set `[FE][lockIP] = 0`
- Set `[FE][lockHashKeyWords] =`

This way the binary used by WebkitPDF is able to access the page to render from "outside".

Warnings on PDF generation page

While generating a PDF sometimes PHP warnings are logged in the webserver's error log. This depends on your TYPO3 and server configuration.

PHP Warning: Cannot modify header information - headers already sent by (output started at /path/to/typo3/typo3conf/ext/webkitpdf/pi1/class.tx_webkitpdf_pi1.php:160) in /path/to/typo3/typo3/sysext/cms/tslib/class.tslib_fe.php on line 3229, referer: ...

To fix this, add this TS to the page with the WebKit PDF-plugin:

```
config {
    disableCharsetHeader = 1
    sendCacheHeaders = 0
    additionalHeaders =
}
```

Use a page type to render PDFs

You can use WebkitPDF on a special page type to show a PDF.

```
PDF = PAGE
PDF.typeNum = 250
PDF.config {
    disableAllHeaderCode = 1
    disableCharsetHeader = 1
    disablePrefixComment = 1
}

PDF.10 < plugin.tx_webkitpdf_pil
PDF.10 {
    customScriptPath = /usr/bin/
    urls {
        typolink.parameter.data = TSFE:id
        typolink.returnLast = url
        dataWrap = {getIndpEnv:TYPO3_REQUEST_HOST}/|
    }
    staticFileName.data = page:title
    staticFileName.wrap = |.pdf
}
```

This way, all calls to &type=250 result in a PDF.
Now you can put a link to all your pages:

```
page.1 = TEXT
page.1 {
    typolink {
        parameter.data = TSFE:id
        additionalParams = &type=250
    }
}
```


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
- 1.1.6
 - Updated binary to version 1.0.0beta2
 - Removed parts from documentation and referred to public manual of the binary instead
 - "scriptParams" have stdWrap too now