Web Page to PDF Conversion with Rmodepdf: Leveraging LuaLaTeX for E-book Reader-friendly Documents

Michal Hoftich

Library, Faculty of Education, CU

July 16, 2024

Introduction

Whad Do We Want to Achieve?

From various sources (HTML, ePub) create PDF suitable for various outputs:

- e-book readers
- smartphones or tablets
- various print page formats

Why?

- comfortable reading
- archiving
- because we can

What Will I Show?

- usage and configuration of the Rmodepdf command
- HTML processing using LuaXML
- Two packages that simplify automatic typesetting
  - responsive design in LaTeX with the package Responsive
  - prevention of overflow boxes in narrow lines with the package Linebreaker

How Do We Convert HTML to PDF for an E-reader?

Rmodepdf

A script that converts web pages to PDF.

- extracts clean text from articles, without ads and navigation elements on the pages
- allows configuration for individual websites or e-book editions (e.g., Municipal Library in Prague)
- configurable output

`https://github.com/michal-h21/rmodepdf/`

Page with Control Elements and Ads

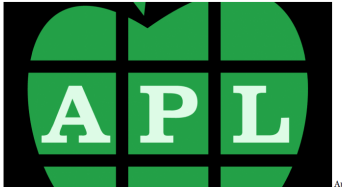
Page in Reader Mode in Firefox

root.cz

Jazyk APL, kombinátory, vláčky a point-free style

Pavel Tišnovský

24-31 minut



Autor: A.Brudz, podle licence: CC-BY

V dalším článku o jazycích z oblasti „array programmingu“ se ještě jednou vrátíme k jazyku APL. Ukážeme si, jak se v nových verzích APL (Dyalog APL) používá elegantní technika nazývaná point-free style nebo též tacit programming.

Obsah

Reader Mode for Scripts

- Reader mode is a feature in web browsers that removes control elements from the page and displays only the article text.
- The projects listed below enable the use of reader mode in scripts.

Readability.js	`https://github.com/mozilla/readability`
Python-readability	`https://github.com/buriy/python-readability`
Rdrview:	`https://github.com/eafer/rdrview`

How Do We Load and Transform HTML Files?

LuaXML contains two libraries for HTML processing and transforming

- the `luaxml-transform` library for converting XML to other formats, such as TEX
  - allows rules for specific elements selected using CSS selectors
- the `luaxml-domobject` library can now load HTML files

Rmodepdf usage

Basic Usage

Rmodepdf accepts multiple URL or filenames as an argument:

```
# process url1 and url2
$ rmodepdf <url1> <url2>
```

Basic Usage

It can also read from the standard input:

```
# process local foo.html passed from the standard input
# "-" will tell rmodepdf to read from stdin
$ cat foo.html | rmodepdf --baseurl foo -
```

Example Output

Title: Jazyk APL, kombinátory, vláčky a point-free style - Root.cz
Author: Pavel Tišnovský
https://www.root.cz/clanky/jazyk-apl-kombinatory-vlacky-a-point-free-style/

Obsah

Print Transformed LaTeX Code

```
# pipe the generated TeX code to foo.tex
$ rmodepdf -p <url> > foo.tex
```

Output File Name

```
# save as foo.pdf
$ rmodepdf -o foo.pdf <url>
```

Choose Page Format and Style

```
# use A4 format for the paper size
# use plain page style
$ rmodepdf -P a4paper -s plain <url>
```

Change Image Directory

```
# save the document as foo.pdf and
# save images in the temp dir
$ rmodepdf -o foo.pdf -i /tmp/img <url>
```

Other Options

-n don't download images

-N don't process LaTeX math in pages

-R don't run Rdrview

-l debug messages log level

Configuration

Loading of the Configuration File

```
# load script.lua as the configuration file
$ rmodepdf -c script.lua <url>
```

Change Settings

```
add_to_config {
  document = {
    preamble_extras = [[
    \setmainfont{Linux Libertine O}
    ]],
  },
  img_convert = {
    -- modify the command used for
    -- conversion of SVG images to PDF
    svg = "cairosvg -o ${dest} -",
  },
}
```

Direct Settings

```
# change settings for the Geometry package
config.document.geometry = "a6paper"
```

Callbacks

Available Callbacks

`preprocess_content`	modify string with the raw HTML before readability and DOM parsing.
`preprocess_dom`	modify DOM object before fetchching of images or handling of MathJax.
`postprocess_dom`	modify DOM after all processing by Rmodepdf.
`postprocess`	late post-processing of the config table.

Example: Print the HTML Code

```
function postprocess_dom(dom)
  print(dom:serialize())
  return dom
end
```

Example: Remove HTML Elements

```
<div class="menu">
... menu contents ...
</div>
```

Example: Remove HTML Elements

```
function postprocess_dom(dom)
  -- Find the menu using a CSS selector
  local menu = dom:query_selector(".menu")

  -- Iterate over the menu elements
  -- and remove each one
  for _, el in ipairs(menu) do
    el:remove_node()
  end

  -- Return the modified DOM
  return dom
end
```

Other Useful LuaXML DOM Functions

`el:get_attribute`	get element attribute
`el:set_attribute`	set element text
`el:get_text`	get text content of the element
`el:get_element_name`	get element name

There are many more functions:

- for traversing the element tree
- for creating new elements

Transformation rules

LuaXML DOM Tranformations

`htmlprocess.add_action`	add a new rule
`htmlprocess.add_custom_action`	process element using Lua
`htmlprocess.reset_actions`	remove rules for the given selector
`%s`	insert transformed contents of the element
`@{<attribute name>}`	insert value of an attribute

Rules Example

```
htmlprocess.reset_actions("figure")
htmlprocess.reset_actions("img")
htmlprocess.add_action("img",
  [[\includegraphics[max width=\textwidth]{@{src}}]])
htmlprocess.add_action("figure", "\n\n \\noindent %s")
```

Templates

Template Basics

- Templates can access variables from the configuration.
- Simple custom syntax

```
# require template
$ rmodepdf -t mytemplate.tex <url>
```

Template Syntax

Variable Printing `@{variablename}`: Prints a variable from the `config` table or its sub-tables.

Loops `_{variablename}loop code/{separator}`: Iterates over array variables, using `%s` placeholders or accessing fields directly.

Conditions `?{variablename}{true}{false}`: Evaluates a condition to insert content based on the presence of variables.

Sample Template Snippet

```
% loop over languages
\usepackage[_{document.languages}%s/{,}]{babel}
% use geometry settings
\usepackage[@{document.geometry}]{geometry}
@{document.preamble_extras}
\begin{document}
% loop over documents
_{pages}
\selectlanguage{@{language}}
% conditionaly print title
?{title}{Title: @{title}}\par}{}
% document contents
@{content}
/{\clearpage}
\end{document}
```

Responsive Design in LaTeX

What is Responsive Design

- flexible structure - adjusting the size of elements on the page to the display device
- media queries - rules applied based on the properties of the display device (screen size, type of display, etc.)

Thanks to these features, the same page code can be well displayed both on a large monitor and on mobile devices.

Page Example on a Large Monitor

Page Example on a Small Screen

The `responsive` Package

A package inspired by responsive design methods for web pages

- adjusting font size to match the display size
- sets basic document dimensions to match the new font size
- typographic scale for font sizes
- media queries

`https://ctan.org/pkg/responsive`

Setting Font Size Based on Display Size

Font size can be set using the command `\setsizes{number of characters per line}`.

```
\begin{minipage}{5cm}
\setsizes{25}

\lipsum[1]

\end{minipage}
```

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis.

Difference in Font Size Based on Number of Characters

`\setsizes{55}`

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

`\setsizes{25}`

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis.

Configuration

Options can be set when calling the package or later using the command `\ResponsiveSetup`.

Important options:

noautomatic do not set font size automatically at the beginning of the document

characters number of characters when automatically setting the font size

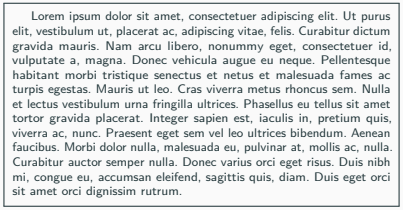
scale typographic scale used for font sizes

lineratio ratio used when calculating line height

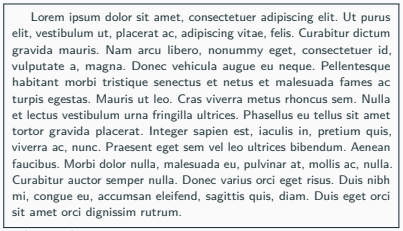
Line Height

Line height can be influenced by the `lineratio` option. The higher its value, the smaller the distance between lines.

`\ResponsiveSetup{lineratio=38}`



`\ResponsiveSetup{lineratio=34}`



https://www.smashingmagazine.com/2020/07/css-techniques-legibility/

CSS Media Query Example

```
body {
  color: green;
}
@media screen and (max-width: 600px) {
    body {
      color: blue;
    }
}
```

Media Queries in LaTeX

Using the `\mediaquery` command, we can test various properties:

- physical page size
- line length
- page orientation

Additional tests can be easily added.

Media Query Example

This example displays fewer characters if the text width is less or equal to 4 cm.

```
\mediaquery{max-textwidth=4cm}
{\setsizes{45}}{\setsizes{60}}
```

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Do Media Queries Make Sense in LaTeX?

- possibly in universal packages
- using different templates for different sizes is easier

The `linebreaker` Package

The `linebreaker` Package

Prevents the occurrence of overfull lines

- Affects only the typesetting of paragraphs where such a line occurs
- If it detects an overfull line in a paragraph, it retypesets it with larger values for `tolerance` and `emergencystretch`.

`https://ctan.org/pkg/linebreaker`

Example

The example document given below creates two pages by using Lua code alone. You will learn how to access TeX's boxes and counters from the Lua side, shipout a page into the PDF file, create horizontal and vertical boxes (hbox and vbox), create new nodes and manipulate the nodes links structure.

Without Linebreaker

The example document given below creates two pages by using Lua code alone. You will learn how to access TeX's boxes and counters from the Lua side, shipout a page into the PDF file, create horizontal and vertical boxes (hbox and vbox), create new nodes and manipulate the nodes links structure.

With Linebreaker

Configuration

Linebreaker can be configured using the `\linebreakersetup` command:

maxcycles	number of attempts to retypeset a paragraph
maxemergencystretch	maximum value of `\emergencystretch`
maxtolerance	maximum value of `tolerance`

```
\linebreakersetup{
maxtolerance = 90,          % default 9999
maxemergencystretch = 1em, % default 3em
maxcycles = 4               % default 30
}
```

Conclusion

Rmodepdf status

- It is still work in progress, so features can change.
- Even if it isn't useful to you, it led to the development of the HTML parser for LuaXML, Linebreaker, and Responsive packages, each of which can be useful independently.

Other useful packages for automatic typesetting

lua-widow-control prevents widows and orphans.

luavlna prevents single chars at end of lines for Czech and Slovak, prevents line breaks in SI units and academic titles.

Thank you for your attention!

michal.h21@gmail.com
www.kodymirus.cz