`pdfscreen.sty` — Manual

C. V. Radhakrishnan

`cvr@river-valley.com`

July 10, 2000

Abstract

`pdfscreen` package helps to redesign the pdf output of your normal documents fit to be read in a computer monitor while retaining the freedom to format it for conventional printing. This has been brought about by redefining the margins and page height/width and related dimensions to fit into that of the computer screen. By changing the options to `print` you can switch the package to format the document in the conventional way as your class file dictates.

Contents

1. Setup

An elaborate manual is not needed for using `pdfscreen`, since it is nothing but an extension of the `hyperref.sty` of Sebastian Rahtz. The primary aim of the package is to change the dimensions of the width and height of the page so as to provide an ideal dimension that is fit for screen viewing rather than printing. As such, all those dimensions that control the page shape are redefined to result the desired screen size. The preamble portion requires the package loading command as given below:

> \usepackage[*screen,panelleft*]{*pdfscreen*}

There is no need to specify the \usepackage{hyperref} with its options, since `hyperref` is loaded by the `pdfscreen`. Unlike previous verions you can load `hyperref.sty` prior to `pdfscreen` with necessary options if you like, if so `pdfscreen` will not reload `hyperref`. The default backend driver for `pdfscreen` is `pdftex`. However, you can specify your backend driver as an option.

It will be nicer if `pdfscreen` is loaded as the last package in the preamble so as to avoid further redefinition of commands that are used by `pdfscreen`.

1.1. Options

The following options are available:

1. `screen` – generates the screen version

2. `print` – generates the print that looks like your `dvi`

3. `panelleft` – navigation panel in the left side

4. `panelright` – navigation panel in the right side

5. `nopanel` – suppresses the panel

6. `paneltoc` – table of contents in the panel. With this option invoked, please do not use \tableofcontents command in the document and `paneltoc` stops as soon as \tableofcontents command is encountered.

7. `sectionbreak` – will introduce pagebreak before a section.

8. `code` – provides commands that can be used to list `verbatim` like listing of program code as found in the LaTeX documentation.

   ```
   \begin{decl}\\
    |\usepackage|\oarg{options}\arg{package}\\
    |\screensize|\Arg{6.25in}\Arg{8in}
   \end{decl}
   ```

> \usepackage[⟨*options*⟩]{⟨*package*⟩}
> \screensize{*6.25in*}{*8in*}

9. **Backend drivers**: `dvips`, `dvipsone`, . . . , `vtex` can be specified as an optional backend driver. `pdftex` is the default.

10. **Color schemes**: There are six color schemes – bluelace, blue, gray, orange, palegreen and chocolate – available for panel and buttons that you can give as an option to the package. Default is blue.

11. **Foreign language support**: Not all the foreign languages are supported. Only 15 European languages are supported at the moment. However, all the language names as you give in the `babel` package can be given here as an option. If the language is not supported, the package will default to English. I may request users to send me the translation of the button text in the navigation panel in your language, if it is not supported.

12. `nocfg` – an option to suppress the configuration file (see subsection 3.4), if you don't want to use its specifications.

1.2. Other parameters to be passed

Few more parameters are to be passed on to the `pdfscreen` to make it more functional. They are:

`\emblema{⟨graphic file name⟩}`

the name of the graphic file that appears on the navigation panel.

`\urlid{⟨URL name⟩}`

The `home page` button in the navigation panel will be linked to the URL.

`\screensize{⟨height⟩}{⟨width⟩}`

This command will facilitate to specify the screen dimensions of the pdf output. No default screen dimensions are available, and therefore the user has to specify it explicitly. There are no restrictions on the screen dimensions. Unlike previous versions, the user is free to choose any dimension. The default width of the panel is 15% of the width of the screen.

`\margins{⟨left⟩}{⟨right⟩}{⟨top⟩}{⟨bottom⟩}`

This command will set the margins of the document. There are no default values for margins and you will have to specify it explicitly in the document preamble.

With `\margins` and `\screensize` explicitly given in the document preamble, `pdfscreen` now obeys whatever screen size and margins the user has specified. This change is brought consequent to the bug report of D. P. Story.

1.3. Typical preamble

A typical document preamble is given below (with which this document is formatted):

```
\documentclass[a4paper,11pt]{article}
\usepackage{xspace,colortbl}

\usepackage[screen,panelleft,gray,paneltoc]{pdfscreen}
\margins{.75in}{.75in}{.75in}{.75in}
\screensize{6.25in}{8in}
\overlay{lightsteelblue.pdf}

\begin{document}
```

1.4. Packages needed to run `pdfscreen`

The following packages are needed for smooth compilation (grab the latest from CTAN:

1. hyperref.sty

2. comment.sty

3. truncate.sty

4. graphicx.sty

5. color.sty

6. colortbl.sty

7. calc.sty

8. amssymb.sty

9. amsbsy.sty

10. shortvrb.sty

11. fancybox.sty

2. Navigation Panel

The design of the navigation panel is left entirely to the imagination of the user. One can create a panel as per his taste. A newer command, \panel has been added and a default panel is also supplied, which is nothing but a box with navigation buttons vertically arranged. There is also a command:

```
\addButton{⟨length⟩}{⟨button text string⟩}
```

which you can use to generate buttons of your choice. Here is an example of how to produce a Next Page button:

```
\Acrobatmenu{NextPage}{\addButton{1.25in}{Next Page}}
```

This will generate the following navigation button

Left Panel	⇒	\usepackage[*screen,panelleft*]{*pdfscreen*}
Right Panel	⇒	\usepackage[*screen,panelright*]{*pdfscreen*}
No Panel	⇒	\usepackage[*screen,nopanel*]{*pdfscreen*}
Portrait	⇒	Change screen size with the command: \screensize{⟨*height*⟩}{⟨*width*⟩}
Square	⇒	Change screen size with the command: \screensize{⟨*height*⟩}{⟨*width*⟩} command
Wide Panel	⇒	\panelwidth=*<dimension>*
Print version	⇒	\usepackage[*print*]{*pdfscreen*}

Figure 1: Different types of panel positioning

Next Page

and clicking this will take you to the next page. In the same way, you can build buttons with images too, for which the command,

```
\imageButton{⟨width⟩}{⟨height⟩}{⟨graphic file name⟩}
```

will be useful.

The TUG image button is generated by:

```
\href{http://www.tug.org}{\imageButton{.5in}{!}{tex.png}}
```

Clicking this button will take you to http://www.tug.org, the TEX Users Group web site.

The navigation panel can be positioned at user's will either to the left side or right side. To see different types of output click the buttons in Figure 1.

Width of the panel can be changed by explicitly giving in the preamble of the document like \panelwidth=⟨*dimension*⟩. The default is 15% of the screen width, however, there is a minimum value of 1in in case 15% of the screen width goes below 1in.

You can define your own panel which is nothing but a vertical box with whatever stuff you want to fit into. The navigation panel of this document is made up of the following code:

```
\panelwidth=1.3in
```

```
\def\panel{\colorbox{panelbackground}
 {\begin{minipage}[t][\paperheight][b]{\panelwidth}
   \centering\null\vspace*{12pt}
 \includegraphics[width=.75in]{univ}\par\vfill
 \href{\@urlid}{\addButton{.85in}{\@Panelhomepagename}}\par\vfill
 \Acrobatmenu{FirstPage}{\addButton{.85in}
   {\FBlack\@Paneltitlepagename}}\par\vfill
 \Acrobatmenu{FirstPage}{\addButton{.2in}
   {\FBlack\scalebox{.8}[1.4]{\btl\btl}}}\hspace{-3pt}
 \Acrobatmenu{PrevPage}{\addButton{.2in}
   {\FBlack\scalebox{.8}[1.4]{\btl}}}\hspace{-3pt}
 \Acrobatmenu{NextPage}{\addButton{.2in}
   {\LBlack\scalebox{.8}[1.4]{\rtl}}}\hspace{-3pt}
 \Acrobatmenu{LastPage}{\addButton{.2in}
   {\LBlack\scalebox{.8}[1.4]{\rtl\rtl}}}\par\vfill
 \Acrobatmenu{GoBack}{\addButton{.85in}
   {\@Panelgobackname}}\par\vfill
 \Acrobatmenu{FullScreen}{\addButton{.85in}{Full Screen}}\par\vfill
 \Acrobatmenu{Close}{\addButton{.85in}{\@Panelclosename}}\par\vfill
 \Acrobatmenu{Quit}{\addButton{.85in}{\@Panelquitname}}\par
 \null\vspace*{12pt}
 \end{minipage}}}
```

3. Other facilities

3.1. Background

The background of the screen area can be overlayed with a graphic file with the command \overlay{⟨*graphic file name*⟩}. Alternatively, you can specify a background color by saying \backgroundcolor{⟨*color*⟩} where color is a predefined color with the commands provided by the color.sty.

The background of the panel can also be provided by a graphic file with the command \paneloverlay{⟨*graphic file*⟩}. If you do not specify an overlay graphic for the panel, the panelbackground color will take effect. You can redefine the panelbackground color to your choice, if you find the default color distasteful, or else can specify the same in the pdfscreen.cfg file. \overlayempty and \paneloverlayempty commands help you to suppress the overlays at any stage.

A new command \changeoverlay has been introduced and a series of small pdf files, each with less than 2 KB file size are offered, so that you can have different overlays with the change of each section unit and it will reset on every tenth section.

You can create your own overlays and modify the \change command with your overlay files.

3.2. Bottom buttons

Bottom footer menu can be invoked with \bottombuttons and can be closed with \nobottombuttons. So also in the academic interest, \topbuttons and \notopbuttons are also available. You can have both the buttons in the same page, though it is bizarre looking.

3.3. Table of contents in the panel

The package option `paneltoc` will allow you to have the table of contents in the navigation panel. In an article, only section headings are shown in the panel toc. Users are requested to use this option with caution, since it is prone to blow up the list and verbatim environments spanning multiple pages. However, a manual intervention with a `\clearpage` command at the appropriate location can ease you to a certain extent.

3.4. Configuration file

You can keep a configuration file called `pdfscreen.cfg` in which you can provide your own translation of button text if your language is unsupported by the package, newer color schemes if you dislike the schemes offered, your URL id, affiliation and division names, date argument, graphic file name of your logo/emblem. I would suggest to use a configuration file, in which you can give all your site specific requirements, that has another advantage of eliminating the clustered look at the preamble of the document. A typical configuration file is supplied with the package.

4. Slides

A slide environment is available which can be entered as

```
\begin{slide}
.
.
.
slide material
.
.
.
\end{slide}
```

This is a box spanning the width and height of the text area, within which the material will be vertically centered.

4.1. Fonts

All the font attributes have been redefined to make them larger than the usual size inside the slide environment. However, if you want to revert to the original size, you will have to add the word `real` before the font size command, i.e., for `\normalsize`, use `\realnormalsize`; for `\large` it is `\reallarge` and so forth.

4.2. Post-processing

The postprocessor *viz.*, `PPower4` can be applied to the pdf generated with this package, so that incremental additions to the pages are possible. `PPower4`

is available at CTAN. You may need Java Virtual Machine running in your system to work with PPower4. I have not tried the T_EXPower package by Stephan Lehmke with pdfscreen, but I would recommend this for effecting incremental builds.

With print option invoked, the slide environment will print as a boxed minipage with another ovalbox adjacent to the slide box. This will prove useful for the audience to record any notes/queries during presentation which they can discuss with the speaker at the end of his session. The 'Notes' that appear in the query box can be redefined to the string you like in your language by the command:

```
\def\notesname{⟨your string⟩}
```

The objectives of the slide option are:

- *to devise a method for easier technical presentation.*

- *to help the mix of mathematical formulae with text and graphics which the present day* WYSIWYG *tools fail to accomplish.*

- *to exploit the platform independence of T_EX so that presentation documents become portable.*

- *to offer the freedom and possibilities of using various backgrounds and other embellishments that a user can imagine to have in his presentation.*

Notes:

5. Page Transition

- You can exploit the page transition facilities in the Acrobat. Specify your choice by using the command \pagedissolve{⟨option⟩}.

- A list of page dissolve options and keys are given in Table 1.

The page dissolve options are taken from the well known book, *Web Publishing with Acrobat/*PDF by Thomas Merz will largely help to know the options for \pagedissolve function.

Keys for page transitions

For some of the transitions, additional parameters may be specified. The code given below results in a split effect with the lines moving horizontally (/H) from the inner parts of the page to the outer parts (/O). The duration of the effect is two seconds (/D):

```
/S /Split /D 2 /Dm /H /M /O
```

All supported parameters for page dissolve, along with the kind of transition on which the parameters may be applied are given in Table 2.

Key	Explanation
/Split	Two lines sweep across the screen to reveal the new page similar to opening a curtain.
/Blinds	Similar to /Split, but with several lines resembling "Venetian blinds"
/Box	A box enlarges from the center of the old page to reveal the new one.
/Wipe	A single line "wipes" across the old page to reveal the new one.
/Dissolve	The old page "dissolves" to reveal the new one.
/Glitter	Similar to /Dissolve, except the effect sweeps from one edge to another.
/R (Replace)	The old page is simply replaced with the new one without any special effect. This is the default.

Table 1: Keys for page transition

Key	Explanation
/D	Duration of the transition effect in seconds (applies to all effects)
/Di (Direction)	Direction of the movement (multiples of 90° only). Values increase in a counterclockwise fashion, 0° points to the right (for /Wipe and /Glitter).
/Dm (Dimension)	Possible values are /H or /V for a horizontal or vertical effect, respectively (for /Split and /Blinds).
/M (Motion)	Specifies whether the effect is performed from the center out or the edges in. Possible values are /I for in and /O for out (for /Split and /Box).

Table 2: Additional parameters for page transitions

6. Bugs and TODO

1. Enumerated and itemized lists spanning across pages create a nasty `missing \item` error. D. P. Story has found out the reason to be the centering command, \begin{center} ... \end{center} code inside the \panel macros. His suggestion to change this to \centering has made a dramatic effect thus eliminating the `missing \item` error. However, if `paneltoc` option is invoked, there is a slight shift in the spacing between table of contents entries in the panel toc is oberved. Still a bug ...

2. `verbatim` environment when spanned across pages (or if a page break occurs amidst a verbatim environment) also had similar bug, resulting in the panel toc items appearing in verbatim mode. A fix has been made again by D. P. Story by putting the verbatim in a vertical box and splitting with \vsplit at the pagebreak. This solved the above problem, but a new problem of loss of colors and font attributes in the second box material after split has errupted.

7. Acknowledgements

navigation panel is due to Kaveh Bazargan of Focal Image Ltd., London and it was at his goading that I started writing this package.

The current version (v1.3) is a complete rewrite and therefore it may be possible that the documents compiled with older versions may break down, though every effort has been made to keep the downward compatibility and the old commands and options are still kept in order not to blow up your documents which runs smoothly with previous versions.

I firmly believe that the code demands further optimization and may have bugs that will show up during exhaustive usage. I request you to send me the bug reports as and when they exhibit their ugly faces.

My special thanks are for all those users who deploy `pdfscreen.sty` for their document preparation, and for their kind words of appreciation.

`pdfscreen` works well with the much used package, `exerquiz.sty` of Donald P. Story (`http://www.math.uakron.edu/˜dpstory/pdf_demos.html`) without clashes. Feedback on the usage of TEXPower with `pdfscreen` is requested from users.

My mail id for contact is `cvr@river-valley.com`.