Manual for the `prosper` class*

Frédéric Goualard
Centrum voor Wiskunde en Informatica
Amsterdam, The Netherlands

Abstract

The `prosper` class permits producing high quality slides; it is also easily extendable. This documentation is meant to be a user manual for the predefined slide styles as well as a technical note describing how to create your own styles.

1 Using the class

LaTeX files using the `prosper` class may be eventually translated into two different formats:

- the Adobe® *PostScript*™ format for printing transparencies;
- the Adobe® *Portable Document Format* (PDF) for displaying slides on computers with Acrobat® Reader in full-screen mode.

When translated into PDF files, `prosper` slides benefit from additional possibilities such as transition effects between slides and incremental displaying of a slide with several animation effects. The currently supported transitions are:

- **`Split`**: two lines sweep across the screen revealing the new slide;
- **`Blinds`**: multiple lines, evenly distributed across the screen, appear and synchronously sweep in the same direction to reveal the new slide;
- **`Box`**: a box sweeps from the center, revealing the new slide;
- **`Wipe`**: a single line sweeps across the screen from one edge to the other, revealing the new slide;
- **`Dissolve`**: the old page image dissolves to reveal the new slide;
- **`Glitter`**: similar to `Dissolve`, except the effect sweeps across the image in a wide band moving from one side of the screen to the other;
- **`Replace`**: the effect is simply to replace the old page with the new page.

Figure 1 presents a bird's-eye view of the structure of a LaTeX file using the `prosper` class.

2 Options of the class

The `prosper` class supports the following options (default options are preceded by a black triangle ▶, while the others are preceded by a black square ■):

■ **`draft.`** The file is compiled in draft mode: figures are replaced by bounding boxes; the caption at the bottom of every slide displays the date and time of the compiling together with the file name;

*This documentation describes version 1.0i of the class. CVS id = $Id: prosper-doc.tex,v 1.7 2001/01/12 08:58:34 exupery Exp

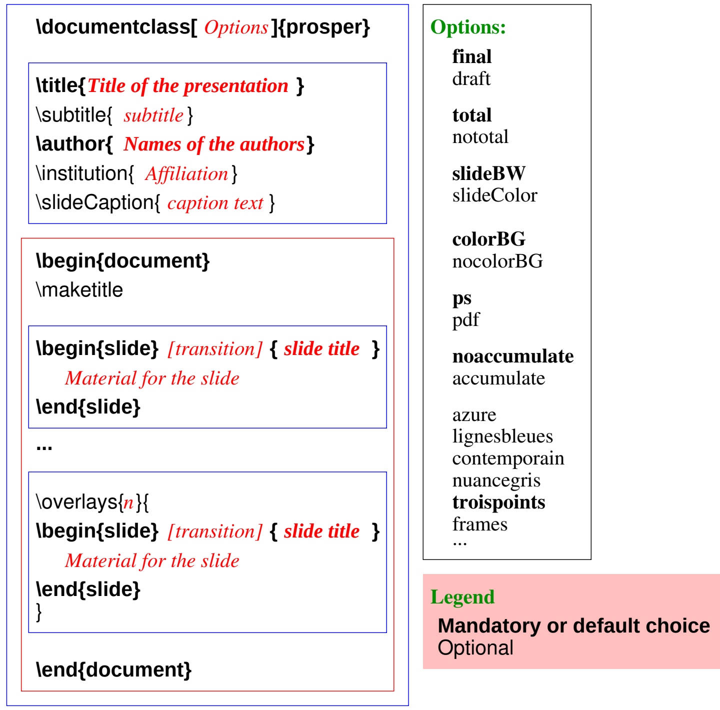

Figure 1: Structure for a LaTeX file using `prosper`

- ▶ **`final`.** The file is compiled in final mode: figures are inserted at their position; the caption on every slide contains the text given (optionally) by the user with the macro `\slideCaption`, except if the macro `\displayVersion` appears in the preamble (in that case, the same caption as in the draft mode is used);
- ■ **`slideColor`.** Slides will use many colors. To be used with caution when the slides are to be printed on a black & white device;
- ▶ **`slideBW`.** Slides will use a restricted set of colors. Should be used whenever the presentation is meant to be printed in black & white;
- ▶ **`total`.** The caption at the bottom of every slide displays the number of the current slide along with the total number of slides;
- ■ **`nototal`.** Only the number of the current slide appears in the caption;
- ▶ **`nocolorBG`.** The background of the slide is white whatever the style may be. It is a good idea to use this option for printing slides in black & white;
- ■ **`colorBG`.** The color of the background depends on the current style;
- ▶ **`ps`.** The LaTeX file is compiled to produce a PostScript™ file for printing;
- ■ **`pdf`.** The LaTeX file is compiled to produce a PDF file for a presentation with a video projector;
- ■ **`accumulate`.** Macros `\onlySlide`, `\untilSlide` and `\fromSlide` interpret their argument in `ps` mode. Note that it is possible to locally modify the option setting by using macros **`\Accumulatetrue`** and **`\Accumulatefalse`**;
- ▶ **`noaccumulate`.** Macros `\onlySlide`, `\untilSlide` and `\fromSlide` do not interpret their argument in `ps` mode;
- ■ **`distiller`.** Use this option if the PostScript® file is to be translated into a PDF file using Adobe® Distiller.

3 Predefined macros and environments

3.1 Macros to appear in the preamble

The `prosper` class (re-)defines some standard macros. Those given hereunder are to be put in the preamble (that is, before `\begin{document}`):

\title. Title of the presentation;

\subtitle. Subtitle of the presentation;

\author. Author(s) of the presentation;

\email. E-mail address of the author(s);

\institution. Name of the institute/company the author(s) come(s) from;

\slideCaption{c}. Caption to be put at the bottom of every slide (name of the event/conference...). The title of the presentation is used as the default caption whenever the author do not override it by providing his own caption by using this macro;

\Logo(x,y){mylogo} or **\Logo{mylogo}**. The logo given by `mylogo` will be put at the position `(x,y)` on each slide (resp. at a default position defined by each slide style). The reference point is bottom left. An example of use is:
`\Logo(2,5){\includegraphics[width=1cm]{irinLOGO.eps}})`

\displayVersion. Displays a draft caption (with the name of the file, the title of the presentation, the name of the author(s), and the date/time of the last LaTeX compiling) instead of the caption defined by the user even when in final mode;

\DefaultTransition{trans}: definition of the default transition mode between slides. By default, the `Replace` mode is used;

\NoFrenchBabelItemize. To be used when loading the babel style with the "french" option in order to have the ability to choose ones own items. The french itemize glue is preserved.

3.2 The `slide` environment

Figure 1 describes the `slide` environment. An optional argument is the transition effect for displaying the slide. The default transition is `R` (Replace).

3.3 Some `itemize` environments

The `Itemize` environment corresponds to the LaTeX `itemize` environment where the text is justified. In `prosper`, the `itemize` environment has been redefined such that text is not justified in it (a better choice for slides).

There also exist an `itemstep` environment where each item is displayed incrementally (in PDF mode).

3.4 Macros to be used out of any `slide` environment

\part[transition]{xx}. Creates a slide only containing the text `xx` vertically and horizontally centered in the font title. The transition `transition`—if given—will be used for this slide.

3.5 Macros that may appear in a `slide` environment

\FontTitle{C}{BW}. Use this macro to change the font/color to be used for slide titles. The first argument is for color slides, the second for black and white ones;

\FontText{C}{BW}. Use this macro to change the font/color to be used for slide text. The first argument is for color slides, the second for black and white ones;

\fontTitle{xx}. Writes its argument using the title font and color;

\fontText{xx}. Writes its argument using the text font and color;

\ColorFoot{col}. The footer is to be written with color `col`;

\PDFtransition{tr}. Uses `tr` as the transition effect from the previous slide to the current slide;

\myitem{lvl}{def}. Defines the item of level `lvl` (where `lvl` may be 1, 2 or 3) to be `def`. By default, it is a green lozenge for all levels. The following code define the items to be 3D bullets of different size and color (the corresponding PostScript™ files are provided in the `img/` directory of the `prosper` distribution):

```
\myitem{1}{\includegraphics[width=.4cm]{red-bullet-on-blue.ps}}
\myitem{2}{\includegraphics[width=.3cm]{green-bullet-on-blue.ps}}
\myitem{3}{\includegraphics[width=.3cm]{yellow-bullet-on-blue.ps}}
```

3.6 Overlays

Overlays add animated effects to slides in PDF mode. They may be used to display a slide incrementally (in several steps), for making appear and disappear some elements on a slide... To use overlays, one has to embed the corresponding `slide` environment into an **\overlays** macro as follows:

```
\overlays{n}{
\begin{slide}{...}
...
\end{slide}}
```

The first argument (`n`) of the `overlays` macro stands for the number of steps composing the animation. The following macros may be used to control what should appear on each slide composing a `n` slides overlay:

- **\fromSlide{p}{mat}**. Puts `mat` on slides `p` through `n`;
- **\onlySlide{p}{mat}**. Puts `mat` on slide `p` only;
- **\untilSlide{p}{mat}**. Puts `mat` on slides `1` through `p`;
- **\FromSlide{p}**. All the material after the occurrence of the macro will be put on slides `p` through `n`;
- **\OnlySlide{p}**. All the material after the occurrence of the macro will be put on slide `p` only;
- **\UntilSlide{p}**. All the material after the occurrence of the macro will be put on slides `1` through `p`.

All those macros are only really meaningful in `pdf` mode; in `ps` mode, they do nothing or interpret their argument, depending on the option `no/accumulate`. Macros `\fromSlide`, `\onlySlide`, and `\untilSlide` accept stared versions which typeset the `mat` material in a zero dimension box (i.e. the position pointer is not moved). The stared versions should be used for replacement purposes. For example, the piece of code:

```
\onlySlide*{1}{\includegraphics{example-1.eps}}%
\onlySlide*{2}{\includegraphics{example-2.eps}}%
\onlySlide*{3}{\includegraphics{example-3.eps}}%
```

would put image `example-1.eps` on the first slide; this image would then be replaced by `example-2.eps` on the second slide, and by `example-3.eps` on the third slide. Note the `%` comment sign at the end of each line: it prevents LaTeX from inserting some space—due to the carriage return—which would induce a slight displacement between each image on the slides.
Important note: keep in mind that macros `\FromSlide`, `\OnlySlide`, `\UntilSlide`, and the un-stared versions of `\fromSlide`, `\onlySlide`, and `\untilSlide` interpret the argument `mat` *for each slide composing the overlay* even if it is not displayed (this is mandatory in order to know the size of the box that needs be reserved for the un-displayed material).

The following macros permit choosing the material to put on a slide depending on the chosen mode (ps or pdf):

- `\`**`PDForPS`**`{`**`ifpdf`**`}{`**`ifps`**`}`. Interprets material `ifpdf` if the chosen mode is `pdf`, otherwise interprets `ifps`;
- `\`**`onlyInPS`**`{`**`mat`**`}`. Interprets material `mat` only if the mode is `ps`;
- `\`**`onlyInPDF`**`{`**`mat`**`}`. Interprets material `mat` only if the mode is `pdf`.

These macros may be used as follows:

```
\overlays{3}{%
\begin{slide}{Example}
\onlySlide*{1}{\includegraphics{example-1.eps}}%
\onlySlide*{2}{\includegraphics{example-2.eps}}%
\onlySlide*{3}{\includegraphics{example-3.eps}}%
\onlyInPS{\includegraphics{example.eps}}%
\end{slide}}
```

This slide will be displayed in three steps with three different figures in `pdf` mode; in `ps` mode, there will be only one slide containing figure `example.eps`.

4 Predefined styles

Figures hereunder present the currently predefined styles with options `colorBG` and `slideColor`. The next section presents some contributed styles by users (available in the `contrib/` directory).

5 Contributed styles

This section may grow as the number of contributed styles increases. Note that the authors of these styles are completely responsible for supporting them.

The persons listed below (alphabetical order) have supported `prosper` by contributing additional great styles. Many thanks to them.

Name	E-mail	Style
Éric Languénou	`Eric.Languenou@irin.univ-nantes.fr`	rico
Guillaume Raschia	`Guillaume.Raschia@irin.univ-nantes.fr`	gyom

6 Warning

The `prosper` slide styles are not bound to provide the same display area. Consequently, using different styles may require some adjustment in the text and graphics positioning.

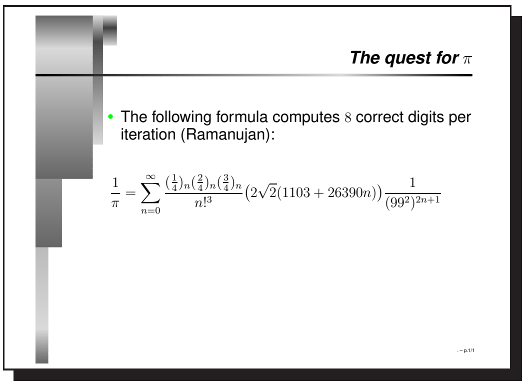

Figure 2: Style `frames` in color

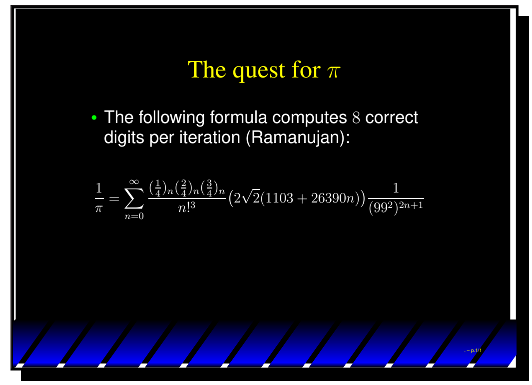

Figure 3: Style `lignesbleues` in color

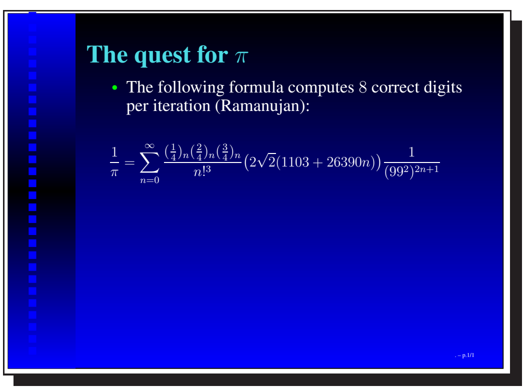

Figure 4: Style `azure` in color

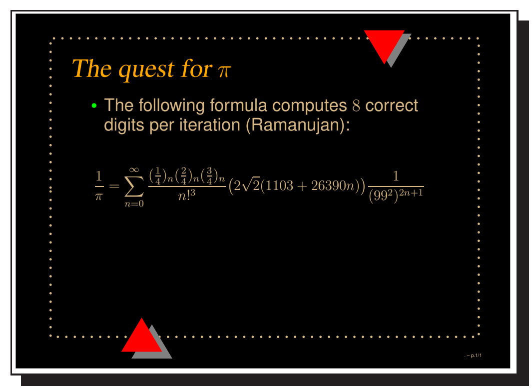

Figure 5: Style `troispoints` in color

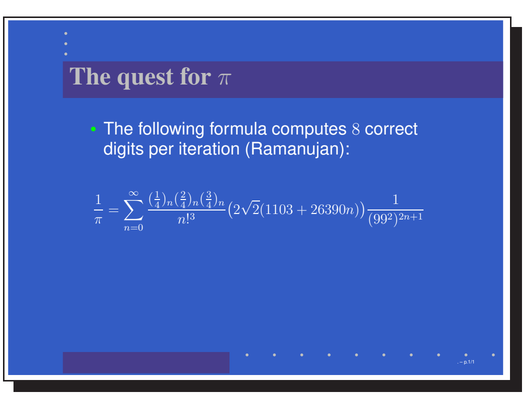

Figure 6: Style `contemporain` in color

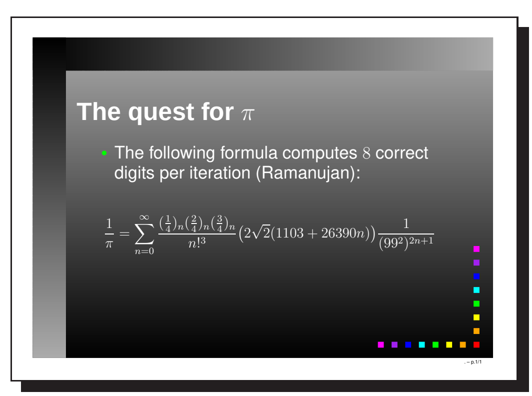

Figure 7: Style `nuancegris` in color

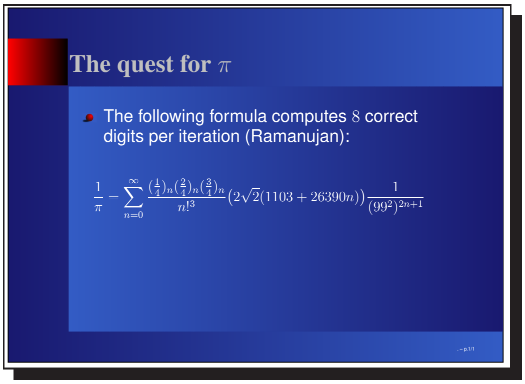

Figure 8: Style `darkblue` in color

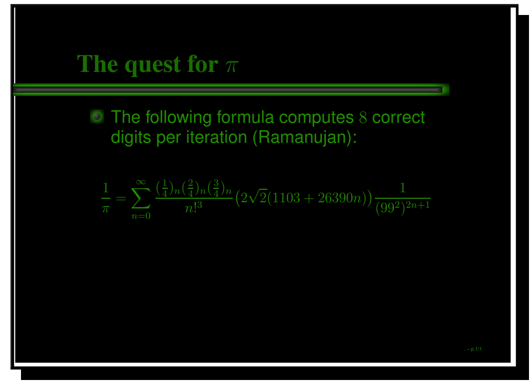

Figure 9: Style `alien glow` in color

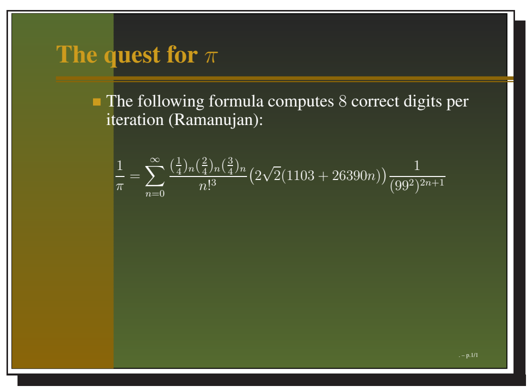

Figure 10: Style `autumn` in color

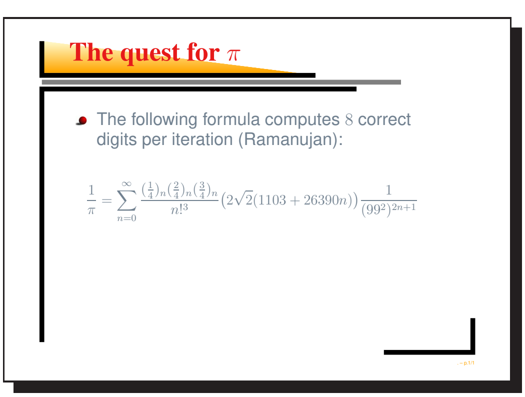

Figure 11: Contributed style `gyom` in color

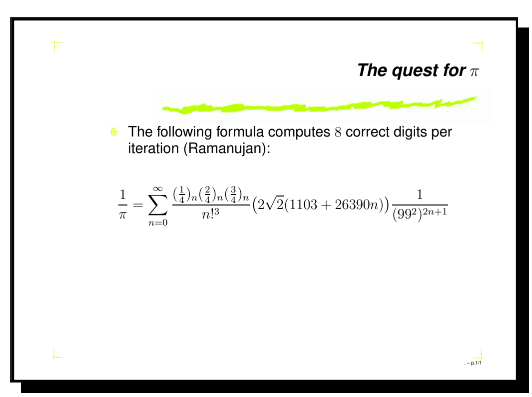

Figure 12: Contributed style `rico` in color

7 The Compilation Process

The compilation process slightly differs depending on the intended use of the slides. It is sketched in Fig. 13. If you plan to print slides on transparencies, you should select the `ps` option and create a PostScript™ file, while if you want to display them with a computer and an overhead projector, you should select the `pdf` option and create a PDF file from the PostScript™ file. Translation of a PostScript™ file into a PDF file is done by the program `ps2pdf` included in the GhostScript distribution.

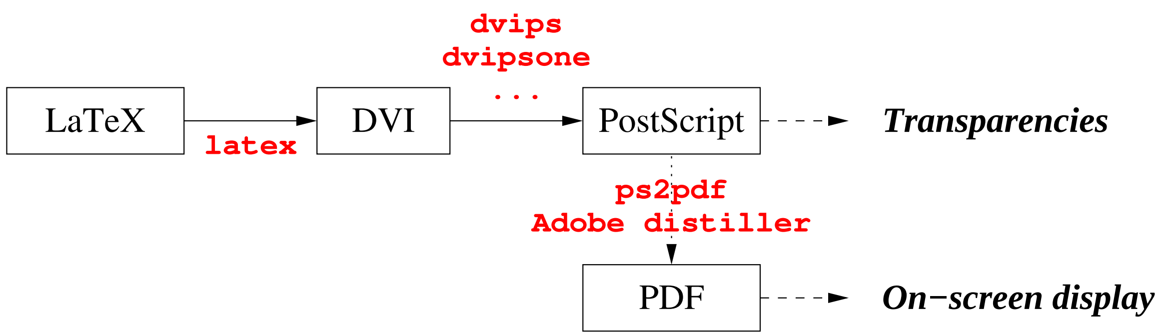

Figure 13: Compilation process

Important note: PDF file should be made resolution independent by using vectorial fonts only (no TEX bitmap fonts). To do so, you have to use a GhostScript version at least equal to 6.0. You also need to create a `.dvipsrc` file in your home directory with the following lines:

```
p +psfonts.cmz
p +psfonts.amz
```

Last, `prosper` styles have been devised to be used with A4 European paper format. Consequently, you will have to instruct GhostScript to use the appropriate format by defining the **`GS_OPTIONS`** environment variable to `"-sPAPERSIZE=a4"`. If you use bash as your main shell, this is done by adding the line

```
export GS_OPTIONS="-sPAPERSIZE=a4"
```

in your `.bash_profile` file.

You will need Adobe® Acrobat® Reader (`acroread`) to display PDF files. It is available for free on the Adobe® web site. Acrobat® Reader provides a full-screen mode that is particularly handy for presentations.

8 Devising new slide styles

Devising new `prosper` styles is an easy task provided you know the basics of Van Zandt's `PSTricks` package (refer to *PSTricks: PostScript™ macros for Generic TEX*, User's Guide, Timothy Van Zandt). In order to devise your own style named `foo`, you first have to create a file `PPR`**`foo`**`.sty` which will contain its definition. Refer to predefined styles for some examples and to Section 8.4.

A word of caution: you are free to create a new style by modifying an existing one. In that case, **it is MANDATORY** renaming your file; do NEVER EVER modify a style without renaming it (is that clear enough?). You should also write your name and email address in any of your styles such that users know who to get in touch with when they use the style. Please choose a name for your style that is unique in the `prosper` distribution (with respect to both predefined and contributed styles so far).

Please send slide styles you are proud of. I will add them to the distribution in the `contrib/` directory. Note that I will only consider for addition styles that are indeed original. Modifying the colors or the fonts of an existing one is definitely not sufficient since this can be done by users in their LATEX file by using the provided hooks for customization.

8.1 Predefined tests

The following tests may be used in your style file in order to modify its behaviour according to the active options. The general scheme is:

```
\ifxxxx%
    % The ``then'' part
\else%
    % The ``else'' part
\fi
```

\ifDVItoPS. True when the DVI file will be eventually translated into a PostScript™ file, false when the final target is the PDF format;

\ifisDraft. True when the file is compiled in draft mode;

\ifinColor. True when the option `slideColor` has been chosen;

\ifallPages. True when the option `total` has been chosen;

\ifcolorBG. True when the option `colorBG` has been chosen;

\ifshowVersion. True whenever the macro `\displayVersion` appears in the preamble;

\ifInOverlays. True if the current `slide` environment is embedded into an `overlays` macro.

8.2 Macros to customize or create a style

\slideCaption{cap}. Definition of a caption to appear on every slide;

\PDFCroppingBox{lx ly ux uy}. Definition of a PostScript™ *bounding box* to crop slides for enhancing their appearance on $4/3$ devices such as monitors (only used in PDF mode);

\NewSlideStyle[width]{anchor}{pos}{defin}. Defines a new slide style whose definition is given by the macro `\defin` and whose contents area has width `width` and is put at position `(pos)` with anchor `anchor`. If no width is given, a default width of 11 cm is used;

\LogoPosition{pos}. Default position for a logo if none is given by the user;

\PutLogo. A macro to be put at the end of the macro that defines your own style.

8.3 Lengths

\slideWidth. Defines the width of the text area in the slide. Should not be modified by the user. Corresponds to the firth argument of macro **NewSlideStyle**.

8.4 Example: the `troispoints` style

```
\NeedsTeXFormat{LaTeX2e}[1995/12/01]
\ProvidesPackage{PPRtroispoints}[2000/04/17]
\typeout{`Trois points' style for Prosper ---}
\typeout{(c) 2000 Frederic Goualard, CWI, The Netherlands}
\typeout{CVSId: $Id: prosper-doc.tex,v 1.7 2001/01/12 08:58:34 exupery Exp $}
\typeout{ }

\RequirePackage{amssymb}
% Loading packages necessary to define this slide style.
\IfFileExists{pst-grad}{\RequirePackage{pst-grad}}{\RequirePackage{gradient}}

\newgray{mygrey}{.5}
```

```
\newrgbcolor{mellow}{.847 .72 .525}
\newrgbcolor{orange}{1.00 0.65 0.00}

\FontTitle{%
  \usefont{T1}{ptm}{m}{sl}\fontsize{22pt}{20pt}\selectfont\orange}{%
  \usefont{T1}{ptm}{m}{sl}\fontsize{22pt}{20pt}\selectfont\blue}
\FontText{%
  \mellow\usefont{T1}{phv}{m}{n}\fontsize{14.4pt}{14pt}\selectfont}{%
  \black\usefont{T1}{phv}{m}{n}\fontsize{14.4pt}{14pt}\selectfont}

\ColorFoot{\mellow}

% Positionning of the title of a slide.
\newcommand{\slidetitle}[1]{%
  \rput[l](-0.4,3.7){\parbox{10cm}{\fontTitle{#1}}}
}

% Positionning for a logo
\LogoPosition{-1,-1.1}

% Definition of this style for slides.

\newcommand{\TPFrame}[1]{%
  \ifinColor
  \ifcolorBG
  \psframe[linestyle=none,fillstyle=solid,fillcolor=black](-2,-1.4)(12.5,9)
  \fi
  \fi
  \psframe[linestyle=dotted,dotsep=5pt,linewidth=2pt,linecolor=mellow]%
  (-1,-.5)(11.6,8.3)
  \pspolygon[linestyle=none,fillstyle=solid,%
  fillcolor=mygrey](8.4,8.4)(9.6,8.4)(9,7.4)
  \pspolygon[linestyle=none,fillstyle=solid,%
  fillcolor=red](8.2,8.5)(9.4,8.5)(8.8,7.5)
  \pspolygon[linestyle=none,fillstyle=solid,%
  fillcolor=mygrey](1.4,-1.1)(2.6,-1.1)(2,-.1)
  \pspolygon[linestyle=none,fillstyle=solid,%
  fillcolor=red](1.1,-.9)(2.3,-.9)(1.7,.1)
  \PutLogo % Mandatory
 {#1}}

\NewSlideStyle{t}{5.3,2.9}{TPFrame}
\PDFCroppingBox{10 40 594 800}
\RequirePackage{semhelv}

\endinput
```

9 Copyright information

THE AUTHOR SPECIFICALLY DISCLAIMS ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS ON AN "AS IS" BASIS, AND THE AUTHOR HAS NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

10 The Prosper homepage

The official Prosper homepage is located at Source Forge (tm):

`http://prosper.sourceforge.net/`

You will find there additional information, CVS tarballs, news, up to date distributions of Prosper... If you plan using Prosper on a regular basis, you should consider subscribing to the lists `prosper-users` and `prosper-announce`. Directions to subscribe to them are available on the homepage.

Prosper is also available on the CTAN repository though the version there might sometimes be slightly out of date:

`http://www.ctan.org/tex-archive/macros/latex/contrib/supported/prosper/`

11 Troubleshootings

If you experience some problem when installing or using Prosper, please go first to the Prosper homepage to check whether there is some hint on how to solve it in one of the list archives. If you do not find any answer to your problem, send a mail to the `prosper-users` list. Only in the last resort should you send a mail directly to me. I am a subscriber of this list anyway, so I will certainly answer to you if nobody else does it.

There is also a file `TROUBLESHOOTINGS` in the distribution listing solutions to commonly encountered problems.

Prosper relies on some recent versions of some packages and software (mainly `hyperref` and Aladdin GhostScript). Check the homepage to find links to the required versions.

12 Bugs reports

Bugs are to be reported by filling the appropriate forms available at the Prosper homepage.

13 Getting in touch with the author

You can send me flames or praises at the following address.

`exupery@users.sourceforge.net`

You might find different email addresses scattered in some files of the distribution. However, you should not rely on them to get in touch with me since they are very likely to be out of date at the time you use them.