The pdfrender package

Heiko Oberdiek*
<heiko.oberdiek at googlemail.com>

2016/05/17 v1.4

Abstract

The PDF format has some graphics parameter like line width or text rendering mode. This package provides an interface for setting these parameters.

Contents

1 Documentation **2**
1.1 Usage . . . 2
1.2 Macros . . . 2
1.3 Parameters . . . 2
1.3.1 Details . . . 3
1.4 Color stack . . . 4

2 Implementation **4**
2.1 Look for pdfTeX, its mode and features . . . 6
2.2 Enable color support of LaTeX . . . 8
2.3 Hook into \normalcolor . . . 8
2.4 Declare and setup parameters . . . 13
2.5 Fill and stroke color support . . . 14

3 Test **18**
3.1 Catcode checks for loading . . . 18
3.2 Simple test file . . . 19
3.3 Further tests . . . 20
3.4 Compatibility with plain TeX . . . 22

4 Installation **22**
4.1 Download . . . 22
4.2 Bundle installation . . . 22
4.3 Package installation . . . 23
4.4 Refresh file name databases . . . 23
4.5 Some details for the interested . . . 23

5 Catalogue **24**

6 Acknowledgement **24**

7 References **24**

*Please report any issues at https://github.com/ho-tex/oberdiek/issues

8 History **25**
[2010/01/26 v1.0] 25
[2010/01/27 v1.1] 25
[2010/01/28 v1.2] 25
[2016/05/14 v1.3] 25
[2016/05/17 v1.4] 25

9 Index **25**

1 Documentation

This package pdfrender defines an interface for PDF specific parameters that affects the rendering of graphics or text. The interface and its implementation uses the same technique as package color for color settings. Therefore this package is loaded to enable LaTeX's color interface.

At different places LaTeX uses `\normalcolor` to avoid that header, footer or floats are print in the current color of the main text. `\setgroup@color` is used to start a save box with the color that is set at box saving time. Package pdfrender extends these macros to add its own hooks of its parameters. Therefore LaTeX3 should generalize LaTeX 2_ε's color interface.

1.1 Usage

In LaTeX the package is loaded as normal package. Options are not defined for this package.

```
\usepackage{pdfrender}
```

This package can also be used in plain TeX and even iniTeX:

```
input pdfrender.sty
```

1.2 Macros

`\pdfrender {`⟨*key value list*⟩`}`

The first parameter ⟨*key value list*⟩ contains a list of parameter settings. The key entry is the parameter name. The macro works like `\color` (without optional argument) for color setting.

`\textpdfrender {`⟨*key value list*⟩`} {`⟨*text*⟩`}`

In the same way as `\pdfrender` the first argument specifies the parameters that should be set. This parameter setting affects ⟨*text*⟩ only. Basically it works the same way as `\textcolor` (without optional argument).

1.3 Parameters

The following table shows an overview for the supported parameters and values:

Parameter	Value	Alias
TextRenderingMode	0	Fill
	1	Stroke
	2	FillStroke
	3	Invisible
	4	FillClip
	5	StrokeClip
	6	FillStrokeClip
	7	Clip
LineWidth	*positive number, unit is bp*	*TEX dimen*
LineCapStyle	0	Butt
	1	Round
	2	ProjectingSquare
LineJoinStyle	0	Miter
	1	Round
	2	Bevel
MiterLimit	*positive number*	
Flatness	*number between 0 and 100*	
LineDashPattern	*numbers in square brackets, followed by number, units are bp*	
RenderingIntent	AbsoluteColorimetric RelativeColorimetric Saturation Perceptual	
FillColor		*color specification*
StrokeColor		*color specification*

1.3.1 Details

The description and specification of these parameters are available in the PDF specification [1]. Therefore they are not repeated here.

Value: The values in the second column lists or describe the values that are specified by the PDF specification.

Alias: Instead of magic numbers the package also defines some aliases that can be given as value. Example: `LineCapStyle=Round` has the same effect as `LineCapStyle=1`.

Number: The term *number* means an integer or real number. The real number is given as plain decimal number without exponent. The decimal separator is a period. At least one digit must be present.

LineWidth: As alias a TEX dimen specification can be given. This includes explicit specifications with number and unit, e.g. `LineWidth=0.5pt`. Also LATEX length registers may be used. If ε-TEX's `\dimexpr` is available, then it is automatically added. However package calc is not supported.

FillColor, StrokeColor: Package color or xcolor must be loaded before these options can be used (since version 1.2). LATEX's color support sets both colors at the same time to the same value. However parameter TextRenderingMode offers the value `FillStroke` that makes only sense, if the two color types can

be set separately. If one of the options FillColor or StrokeColor is specified, then also the color is set. For compatibility with the LaTeX color packages (color or xcolor), always both colors must be set. Thus if one of them is not specified, it is taken from the current color.

Both options FillColor and StrokeColor expect a LaTeX color specification as value. Also the optional color model argument is supported. Example:

```
FillColor=yellow,
StrokeColor=[cmyk]{1,.5,0,0}
```

1.4 Color stack

If the pdfTeX version provides color stacks, then each parameter is assigned a page based color stack. The assignment of a stack takes place, when its parameter is set the first time. This avoids the use of color stacks that are not needed.

2 Implementation

```
1 ⟨*package⟩
```

Reload check, especially if the package is not used with LaTeX.

```
2 \begingroup\catcode61\catcode48\catcode32=10\relax%
3   \catcode13=5 % ^^M
4   \endlinechar=13 %
5   \catcode35=6 % #
6   \catcode39=12 % '
7   \catcode44=12 % ,
8   \catcode45=12 % -
9   \catcode46=12 % .
10  \catcode58=12 % :
11  \catcode64=11 % @
12  \catcode123=1 % {
13  \catcode125=2 % }
14  \expandafter\let\expandafter\x\csname ver@pdfrender.sty\endcsname
15  \ifx\x\relax % plain-TeX, first loading
16  \else
17    \def\empty{}%
18    \ifx\x\empty % LaTeX, first loading,
19      % variable is initialized, but \ProvidesPackage not yet seen
20    \else
21      \expandafter\ifx\csname PackageInfo\endcsname\relax
22        \def\x#1#2{%
23          \immediate\write-1{Package #1 Info: #2.}%
24        }%
25      \else
26        \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
27      \fi
28      \x{pdfrender}{The package is already loaded}%
29      \aftergroup\endinput
30    \fi
31  \fi
32 \endgroup%
```

Package identification:

```
33 \begingroup\catcode61\catcode48\catcode32=10\relax%
34   \catcode13=5 % ^^M
35   \endlinechar=13 %
36   \catcode35=6 % #
37   \catcode39=12 % '
38   \catcode40=12 % (
39   \catcode41=12 % )
```

```
  \catcode44=12 % ,
  \catcode45=12 % -
  \catcode46=12 % .
  \catcode47=12 % /
  \catcode58=12 % :
  \catcode64=11 % @
  \catcode91=12 % [
  \catcode93=12 % ]
  \catcode123=1 % {
  \catcode125=2 % }
  \expandafter\ifx\csname ProvidesPackage\endcsname\relax
    \def\x#1#2#3[#4]{\endgroup
      \immediate\write-1{Package: #3 #4}%
      \xdef#1{#4}%
    }%
  \else
    \def\x#1#2[#3]{\endgroup
      #2[{#3}]%
      \ifx#1\@undefined
        \xdef#1{#3}%
      \fi
      \ifx#1\relax
        \xdef#1{#3}%
      \fi
    }%
  \fi
\expandafter\x\csname ver@pdfrender.sty\endcsname
\ProvidesPackage{pdfrender}%
  [2016/05/17 v1.4 Access to some PDF graphics parameters (HO)]%
\begingroup\catcode61\catcode48\catcode32=10\relax%
  \catcode13=5 % ^^M
  \endlinechar=13 %
  \catcode123=1 % {
  \catcode125=2 % }
  \catcode64=11 % @
  \def\x{\endgroup
    \expandafter\edef\csname PdfRender@AtEnd\endcsname{%
      \endlinechar=\the\endlinechar\relax
      \catcode13=\the\catcode13\relax
      \catcode32=\the\catcode32\relax
      \catcode35=\the\catcode35\relax
      \catcode61=\the\catcode61\relax
      \catcode64=\the\catcode64\relax
      \catcode123=\the\catcode123\relax
      \catcode125=\the\catcode125\relax
    }%
  }%
\x\catcode61\catcode48\catcode32=10\relax%
\catcode13=5 % ^^M
\endlinechar=13 %
\catcode35=6 % #
\catcode64=11 % @
\catcode123=1 % {
\catcode125=2 % }
\def\TMP@EnsureCode#1#2{%
  \edef\PdfRender@AtEnd{%
    \PdfRender@AtEnd
    \catcode#1=\the\catcode#1\relax
  }%
  \catcode#1=#2\relax
}
\TMP@EnsureCode{10}{12}% ^^J
```

```
\TMP@EnsureCode{36}{3}% $
\TMP@EnsureCode{39}{12}% '
\TMP@EnsureCode{40}{12}% (
\TMP@EnsureCode{41}{12}% )
\TMP@EnsureCode{42}{12}% *
\TMP@EnsureCode{43}{12}% +
\TMP@EnsureCode{44}{12}% ,
\TMP@EnsureCode{45}{12}% -
\TMP@EnsureCode{46}{12}% .
\TMP@EnsureCode{47}{12}% /
\TMP@EnsureCode{58}{12}% :
\TMP@EnsureCode{59}{12}% ;
\TMP@EnsureCode{60}{12}% <
\TMP@EnsureCode{62}{12}% >
\TMP@EnsureCode{63}{12}% ?
\TMP@EnsureCode{91}{12}% [
\TMP@EnsureCode{93}{12}% ]
\TMP@EnsureCode{94}{7}% ^ (superscript)
\TMP@EnsureCode{96}{12}% `
\TMP@EnsureCode{124}{12}% |
\def\PdfRender@AtEndHook{}
\expandafter\def\expandafter\PdfRender@AtEnd\expandafter{%
  \expandafter\PdfRender@AtEndHook
  \PdfRender@AtEnd
  \endinput
}
```

2.1 Look for pdfTEX, its mode and features

\PdfRender@newif

```
\def\PdfRender@newif#1{%
  \expandafter\edef\csname PdfRender@#1true\endcsname{%
    \let
    \expandafter\noexpand\csname ifPdfRender@#1\endcsname
    \noexpand\iftrue
  }%
  \expandafter\edef\csname PdfRender@#1false\endcsname{%
    \let
    \expandafter\noexpand\csname ifPdfRender@#1\endcsname
    \noexpand\iffalse
  }%
  \csname PdfRender@#1false\endcsname
}
```

\ifPdfRender@Stack

```
\PdfRender@newif{Stack}
```

\ifPdfRender@Match

```
\PdfRender@newif{Match}
```

\PdfRender@RequirePackage

```
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname RequirePackage\endcsname\relax
  \def\PdfRender@RequirePackage#1[#2]{%
    \expandafter\def\expandafter\PdfRender@AtEndHook\expandafter{%
      \PdfRender@AtEndHook
      \ltx@ifpackagelater{#1}{#2}{}{%
        \@PackageWarningNoLine{pdfrender}{%
          You have requested version\MessageBreak
          '#2' of package '#1',\MessageBreak
          but only version\MessageBreak
          '\csname ver@#1.\ltx@pkgextension\endcsname'\MessageBreak
```

```
        is available%
      }%
    }%
  }%
  \input #1.sty\relax
 }%
\else
 \let\PdfRender@RequirePackage\RequirePackage
\fi
```

Luatex compatibility

```
\ifx\pdfextension\@undefined\else
 \PdfRender@RequirePackage{luatex85}[2016/01/01]
\fi
\PdfRender@RequirePackage{ifpdf}[2010/01/28]
\PdfRender@RequirePackage{infwarerr}[2007/09/09]
\PdfRender@RequirePackage{ltxcmds}[2010/01/28]
\ifpdf
 \ltx@IfUndefined{pdfcolorstackinit}{%
  \@PackageWarning{pdfrender}{%
   Missing \string\pdfcolorstackinit
  }%
 }{%
  \PdfRender@Stacktrue
 }%
 \ltx@IfUndefined{pdfmatch}{%
  \@PackageInfoNoLine{pdfrender}{%
   \string\pdfmatch\ltx@space not found. %
   Therefore the values\MessageBreak
   of some parameters are not validated%
  }%
 }{%
  \PdfRender@Matchtrue
 }%
\else
 \@PackageWarning{pdfrender}{%
  Missing pdfTeX in PDF mode%
 }%
 \ltx@IfUndefined{newcommand}{%
```

\pdfrender

```
  \def\pdfrender#1{}%
```

\textpdfrender

```
  \long\def\textpdfrender#1#2{#2}%
 }{%
```

\pdfrender

```
  \newcommand*{\pdfrender}[1]{}%
```

\textpdfrender

```
  \newcommand{\textpdfrender}[2]{#2}%
 }%
 \expandafter\PdfRender@AtEnd
\fi%
```

2.2 Enable color support of LaTeX

```
\ltx@ifpackageloaded{color}{}{%
  \def\color@setgroup{\begingroup\set@color}%
  \let\color@begingroup\begingroup
  \def\color@endgroup{\endgraf\endgroup}%
  \def\color@hbox{\hbox\bgroup\color@begingroup}%
  \def\color@vbox{\vbox\bgroup\color@begingroup}%
  \def\color@endbox{\color@endgroup\egroup}%
  \ltx@ifundefined{bgroup}{%
    \let\bgroup={\let\egroup=}%
  }{}%
  \ltx@ifundefined{endgraf}{%
    \let\endgraf=\par
  }{}%
}
```

2.3 Hook into \normalcolor

The problem is that packages color and xcolor each overwrite \normalcolor. For example, after the package loading order color, pdfrender and xcolor the patched version of \normalcolor is overwritten by package xcolor. Also using \AtBeginDocument for patching is not enough. If package hyperref is loaded later, it might load package color using \AtBeginDocument.

\PdfRender@NormalColorHook

```
\def\PdfRender@NormalColorHook{}
```

\PdfRender@ColorSetGroupHook

```
\def\PdfRender@ColorSetGroupHook{}
```

\PdfRender@TestBox

```
\def\PdfRender@TestBox#1{%
  \setbox0=\color@hbox#1\color@endbox
}
```

\PdfRender@PatchNormalColor

```
\def\PdfRender@PatchNormalColor{%
  \ltx@ifundefined{normalcolor}{%
    \gdef\normalcolor{\PdfRender@NormalColorHook}%
  }{%
    \begingroup
      \def\PdfRender@NormalColorHook{\let\PdfRender@temp=Y}%
      \PdfRender@TestBox{%
        \let\set@color\relax
        \normalcolor
        \ifx\PdfRender@temp Y%
        \else
          \ltx@GlobalAppendToMacro\normalcolor{%
            \PdfRender@NormalColorHook
          }%
        \fi
      }%
    \endgroup
  }%
  \ifx\@nodocument\relax
    \global\let\PdfRender@PatchNormalColor\relax
  \fi
}%
```

\PdfRender@PatchColorSetGroup

```
\def\PdfRender@PatchColorSetGroup{%
  \begingroup
```

```
    \def\PdfRender@ColorSetGroupHook{\let\PdfRender@temp=Y}%
    \PdfRender@TestBox{%
      \let\set@color\relax
      \color@setgroup\color@endgroup
      \ifx\PdfRender@temp Y%
      \else
        \ltx@GlobalAppendToMacro\color@setgroup{%
          \PdfRender@ColorSetGroupHook
        }%
      \fi
    }%
  \endgroup
  \ifx\@nodocument\relax
    \global\let\PdfRender@PatchColorSetGroup\relax
  \fi
}%
```

\PdfRender@PatchColor

```
\def\PdfRender@PatchColor{%
  \PdfRender@PatchNormalColor
  \PdfRender@PatchColorSetGroup
}

\PdfRender@PatchColor
\ltx@IfUndefined{AtBeginDocument}{}{%
  \AtBeginDocument{\PdfRender@PatchColor}%
}
```

\AfterPackage is provided by package scrlfile.

```
\ltx@IfUndefined{AfterPackage}{%
}{%
  \AfterPackage{color}{\PdfRender@PatchColor}%
  \AfterPackage{xcolor}{\PdfRender@PatchColor}%
  \AfterPackage{etoolbox}{%
    \AfterEndPreamble{\PdfRender@PatchColor}%
  }%
}%
```

\AfterEndPreamble is provided by package etoolbox.

```
\ltx@IfUndefined{AfterEndPreamble}{%
}{%
  \AfterEndPreamble{\PdfRender@PatchColor}%
}%
\PdfRender@RequirePackage{kvsetkeys}[2010/01/28]
```

\PdfRender@texorpdfstring

```
\def\PdfRender@texorpdfstring{%
  \ltx@IfUndefined{texorpdfstring}\ltx@firstoftwo\texorpdfstring
}
```

\pdfrender

```
\ltx@IfUndefined{DeclareRobustCommand}%
\ltx@firstoftwo\ltx@secondoftwo
{%
  \def\pdfrender#1%
}{%
  \newcommand{\pdfrender}{}%
  \DeclareRobustCommand*{\pdfrender}[1]%
}%
{%
  \PdfRender@texorpdfstring{%
    \PdfRender@PatchNormalColor
    \global\let\PdfRender@FillColor\ltx@empty
```

```
    \global\let\PdfRender@StrokeColor\ltx@empty
    \kvsetkeys{PDFRENDER}{#1}%
    \PdfRender@SetColor
  }{}%
}
```

\textpdfrender

```
\ltx@IfUndefined{DeclareRobustCommand}%
\ltx@firstoftwo\ltx@secondoftwo
{%
  \long\def\textpdfrender#1#2%
}{%
  \newcommand{\textpdfrender}{}%
  \DeclareRobustCommand{\textpdfrender}[2]%
}%
{%
  \PdfRender@texorpdfstring{%
    \begingroup
      \pdfrender{#1}%
      #2%
    \endgroup
  }{#2}%
}
```

\ifPdfRender@Values

```
\PdfRender@newif{Values}
```

\PdfRender@NewClassValues

```
\def\PdfRender@NewClassValues#1#2#3#4{%
  \PdfRender@Valuestrue
  \PdfRender@NewClass{#1}{#2}{#3}{#4}{}%
}
```

\PdfRender@NewClass

```
\def\PdfRender@NewClass#1#2#3#4#5{%
  \PdfRender@newif{Active#1}%
  \expandafter\def\csname PdfRender@Default#1\endcsname{#2}%
  \expandafter\let\csname PdfRender@Current#1\expandafter\endcsname
     \csname PdfRender@Default#1\endcsname
  \ifPdfRender@Stack
    \expandafter\edef\csname PdfRender@Init#1\endcsname{%
      \global\chardef
      \expandafter\noexpand\csname PdfRender@Stack#1\endcsname=%
         \noexpand\pdfcolorstackinit page direct{%
           \noexpand#3%
           \expandafter\noexpand\csname PdfRender@Default#1\endcsname
         }\relax
      \noexpand\@PackageInfo{pdfrender}{%
        New color stack '#1' = \noexpand\number
        \expandafter\noexpand\csname PdfRender@Stack#1\endcsname
      }%
      \gdef\expandafter\noexpand\csname PdfRender@Init#1\endcsname{}%
    }%
    \expandafter\edef\csname PdfRender@Set#1\endcsname{%
      \expandafter\noexpand\csname PdfRender@Init#1\endcsname
      \noexpand\pdfcolorstack
      \expandafter\noexpand\csname PdfRender@Stack#1\endcsname
      push{%
        #3{\expandafter\noexpand\csname PdfRender@Current#1\endcsname}%
      }%
      \noexpand\aftergroup
      \expandafter\noexpand\csname PdfRender@Reset#1\endcsname
```

```
    }%
    \expandafter\edef\csname PdfRender@Reset#1\endcsname{%
      \expandafter\noexpand\csname PdfRender@Init#1\endcsname
      \noexpand\pdfcolorstack
      \expandafter\noexpand\csname PdfRender@Stack#1\endcsname
      pop\relax
    }%
  \else
    \expandafter\edef\csname PdfRender@Set#1\endcsname{%
      \noexpand\pdfliteral direct{%
        #3{\expandafter\noexpand\csname PdfRender@Current#1\endcsname}%
      }%
      \noexpand\aftergroup
      \expandafter\noexpand\csname PdfRender@Reset#1\endcsname
    }%
    \expandafter\edef\csname PdfRender@Reset#1\endcsname{%
      \noexpand\pdfliteral direct{%
        #3{\expandafter\noexpand\csname PdfRender@Current#1\endcsname}%
      }%
    }%
  \fi
  \expandafter\edef\csname PdfRender@Normal#1\endcsname{%
    \let
    \expandafter\noexpand\csname PdfRender@Current#1\endcsname
    \expandafter\noexpand\csname PdfRender@Default#1\endcsname
    \noexpand\PdfRender@Set{#1}%
  }%
   \expandafter\ltx@GlobalAppendToMacro\expandafter\PdfRender@NormalCol-
orHook
  \expandafter{%
    \csname PdfRender@Normal#1\endcsname
  }%
  \ltx@GlobalAppendToMacro\PdfRender@ColorSetGroupHook{%
    \PdfRender@Set{#1}%
  }%
  \ifPdfRender@Values
    \kv@parse@normalized{#4}{%
      \expandafter\let\csname PdfRender@#1@\kv@key\endcsname\kv@key
      \ifx\kv@value\relax
      \else
        \expandafter\let\csname PdfRender@#1@\kv@value\endcsname\kv@key
      \fi
      \ltx@gobbletwo
    }%
    \PdfRender@define@key{PDFRENDER}{#1}{%
      \global\csname PdfRender@Active#1true\endcsname
      \def\PdfRender@Current{##1}%
      \PdfRender@SetValidateValues{#1}%
    }%
    \PdfRender@Valuesfalse
  \else
    \PdfRender@define@key{PDFRENDER}{#1}{%
      \global\csname PdfRender@Active#1true\endcsname
      \expandafter\def\csname PdfRender@Current#1\endcsname{##1}%
      \ltx@IfUndefined{PdfRender@PostProcess#1}{%
      }{%
        \csname PdfRender@PostProcess#1\endcsname
      }%
      \PdfRender@SetValidate{#1}{#4}{#5}%
    }%
  \fi
}%
```

\PdfRender@define@key

```
\ltx@IfUndefined{define@key}{%
  \def\PdfRender@define@key#1#2{%
    \expandafter\def\csname KV@#1@#2\endcsname##1%
  }%
}{%
  \let\PdfRender@define@key\define@key
}
```

\PdfRender@Set

```
\def\PdfRender@Set#1{%
  \csname ifPdfRender@Active#1\endcsname
    \csname PdfRender@Set#1\expandafter\endcsname
  \fi
}
```

\PdfRender@Reset

```
\def\PdfRender@Reset#1{%
  \csname ifPdfRender@Active#1\endcsname
    \csname PdfRender@Reset#1\expandafter\endcsname
  \fi
}
```

\PdfRender@ErrorInvalidValue

```
\def\PdfRender@ErrorInvalidValue#1{%
  \PackageError{pdfrender}{%
    Ignoring parameter setting for '#1'\MessageBreak
    because of invalid value %
    '\csname PdfRender@Current#1\endcsname'%
  }\@ehc
  \expandafter\let\csname PdfRender@Current#1\endcsname\ltx@empty
}%
```

\PdfRender@SetValidate

```
\ifPdfRender@Match
  \def\PdfRender@SetValidate#1#2#3{%
    \ifnum\pdfmatch{^(#2)$}{\csname PdfRender@Current#1\endcsname}=1 %
      \csname PdfRender@Set#1\expandafter\endcsname
    \else
      \PdfRender@ErrorInvalidValue{#1}%
    \fi
  }%
\else
  \def\PdfRender@SetValidate#1#2#3{%
    \expandafter\let\expandafter\PdfRender@Current
    \csname PdfRender@Current#1\endcsname
    #3%
    \ifx\PdfRender@Current\@empty
      \PdfRender@ErrorInvalidValue{#1}%
    \else
      \csname PdfRender@Set#1\expandafter\endcsname
    \fi
  }%
\fi
```

\PdfRender@SetValidateValues

```
\def\PdfRender@SetValidateValues#1{%
  \ltx@IfUndefined{PdfRender@#1@\PdfRender@Current}{%
    \expandafter\let\csname PdfRender@Current#1\endcsname
                \PdfRender@Current
    \PdfRender@ErrorInvalidValue{#1}%
  }{%
```

```
    \expandafter\edef\csname PdfRender@Current#1\endcsname{%
      \csname PdfRender@#1@\PdfRender@Current\endcsname
    }%
    \csname PdfRender@Set#1\endcsname
  }%
}
```

\PdfRender@OpValue

```
\def\PdfRender@OpValue#1#2{#2\ltx@space#1}%
```

\PdfRender@OpName

```
\def\PdfRender@OpName#1#2{/#2\ltx@space#1}%
```

2.4 Declare and setup parameters

```
\PdfRender@NewClassValues{TextRenderingMode}%
                  {0}%
                  {\PdfRender@OpValue{Tr}}{%
  0=Fill,%
  1=Stroke,%
  2=FillStroke,%
  3=Invisible,%
  4=FillClip,%
  5=StrokeClip,%
  6=FillStrokeClip,%
  7=Clip,%
}%
\PdfRender@NewClass{LineWidth}{1}{\PdfRender@OpValue{w}}{%
  [0-9]+\string\.?[0-9]*|\string\.[0-9]+%
}{}%
\ltx@IfUndefined{dimexpr}{%
  \def\PdfRender@dimexpr{}%
}{%
  \let\PdfRender@dimexpr\dimexpr
}
\def\PdfRender@PostProcessLineWidth{%
  \begingroup
  \afterassignment\PdfRender@@PostProcessLineWidth
  \dimen0=\PdfRender@dimexpr\PdfRender@CurrentLineWidth bp %
  \PdfRender@let\PdfRender@relax\PdfRender@relax
}
\let\PdfRender@let\let
\let\PdfRender@relax\relax
\def\PdfRender@@PostProcessLineWidth#1\PdfRender@let{%
  \ifx\\#1\\%
    \endgroup
  \else
    \dimen0=.996264\dimen0 % 72/72.27
    \edef\x{\endgroup
      \def\noexpand\PdfRender@CurrentLineWidth{%
        \strip@pt\dimen0%
      }%
    }%
    \expandafter\x
  \fi
}
\PdfRender@NewClassValues{LineCapStyle}{0}{\PdfRender@OpValue{J}}{%
  0=Butt,%
  1=Round,%
  2=ProjectingSquare,%
}%
\PdfRender@NewClassValues{LineJoinStyle}{0}{\PdfRender@OpValue{j}}{%
```

```
  0=Miter,%
  1=Round,%
  2=Bevel,%
}%
\PdfRender@NewClass{MiterLimit}{10}{\PdfRender@OpValue{M}}{%
  [0-9]*[1-9][0-9]*\string\.?[0-9]*|%
  [0-9]*\string\.?[0-9]*[1-9][0-9]*%
}{}%
\PdfRender@NewClass{Flatness}{0}{\PdfRender@OpValue{i}}{%
  100(\string\.0*)?|[0-9][0-9](\string\.[0-9]*)?|\string\.[0-9]+%
}{}%
\PdfRender@NewClass{LineDashPattern}{[]0}{\PdfRender@OpValue{d}}{%
  \string\[%
  ( ?([0-9]+\string\.?[0-9]*|\string\.[0-9]+) ?)*%
  \string\] ?%
  ([0-9]+\string\.?[0-9]*|\string\.[0-9]+)%
}{}%
\PdfRender@NewClassValues{RenderingIntent}%
                    {RelativeColorimetric}%
                    {\PdfRender@OpName{ri}}{%
  AbsoluteColorimetric,%
  RelativeColorimetric,%
  Saturation,%
  Perceptual,%
}%
```

2.5 Fill and stroke color support

```
\PdfRender@define@key{PDFRENDER}{FillColor}{%
  \begingroup
    \def\PdfRender@Color{#1}%
    \ifx\PdfRender@Color\ltx@empty
      \global\let\PdfRender@FillColor\ltx@empty
    \else
      \PdfRender@ColorAvailable{%
        \PdfRender@TestBox{%
          \expandafter\PdfRender@TryColor\PdfRender@Color\ltx@empty
          \PdfRender@GetFillColor
          \ifx\PdfRender@FillColor\ltx@empty
            \@PackageWarning{pdfrender}{%
              Cannot extract fill color\MessageBreak
              from value '#1'%
            }%
          \fi
        }%
      }%
    \fi
  \endgroup
}
\PdfRender@define@key{PDFRENDER}{StrokeColor}{%
  \begingroup
    \def\PdfRender@Color{#1}%
    \ifx\PdfRender@Color\ltx@empty
      \global\let\PdfRender@StrokeColor\ltx@empty
    \else
      \PdfRender@ColorAvailable{%
        \PdfRender@TestBox{%
          \expandafter\PdfRender@TryColor\PdfRender@Color\ltx@empty
          \PdfRender@GetStrokeColor
          \ifx\PdfRender@StrokeColor\ltx@empty
            \@PackageWarning{pdfrender}{%
              Cannot extract stroke color\MessageBreak
              from value '#1'%
```

```
        }%
      \fi
    }%
  }%
 \fi
 \endgroup
}
```

\PdfRender@ColorAvailable

```
\def\PdfRender@ColorAvailable{%
 \@ifundefined{set@color}{%
  \@PackageError{pdfrender}{%
   Ignoring color options, because neither\MessageBreak
   package 'color' nor package 'xcolor' is loaded%
  }\@ehc
  \global\let\PdfRender@ColorAvailable\ltx@gobble
 }{%
  \global\let\PdfRender@ColorAvailable\ltx@firstofone
 }%
 \PdfRender@ColorAvailable
}
```

\PdfRender@TryColor

```
\def\PdfRender@TryColor{%
 \@ifnextchar[\color\PdfRender@@TryColor
}
```

\PdfRender@@TryColor

```
\def\PdfRender@@TryColor#1\ltx@empty{%
 \expandafter\color\expandafter{\PdfRender@Color}%
}
```

\PdfRender@SetColor

```
\def\PdfRender@SetColor{%
 \chardef\PdfRender@NeedsCurrentColor=0 %
 \ifx\PdfRender@FillColor\ltx@empty
  \ifx\PdfRender@StrokeColor\ltx@empty
  \else
   \edef\PdfRender@CurrentColor{%
    \noexpand\PdfRender@FillColor\ltx@space\PdfRender@StrokeColor
   }%
   \chardef\PdfRender@NeedsCurrentColor=1 %
  \fi
 \else
  \ifx\PdfRender@StrokeColor\ltx@empty
   \edef\PdfRender@CurrentColor{%
    \PdfRender@FillColor\ltx@space\noexpand\PdfRender@StrokeColor
   }%
   \chardef\PdfRender@NeedsCurrentColor=2 %
  \else
   \edef\current@color{%
    \PdfRender@FillColor\ltx@space\PdfRender@StrokeColor
   }%
   \set@color
  \fi
 \fi
 \ifnum\PdfRender@NeedsCurrentColor=1 %
  \PdfRender@GetFillColor
  \ifx\PdfRender@FillColor\ltx@empty
   \@PackageWarning{pdfrender}{%
    Cannot extract current fill color%
   }%
  \else
```

```
      \edef\current@color{\PdfRender@CurrentColor}%
      \set@color
    \fi
  \else
    \ifnum\PdfRender@NeedsCurrentColor=2 %
      \PdfRender@GetStrokeColor
      \ifx\PdfRender@StrokeColor\ltx@empty
        \@PackageWarning{pdfrender}{%
          Cannot extract current stroke color%
        }%
      \else
        \edef\current@color{\PdfRender@CurrentColor}%
        \set@color
      \fi
    \fi
  \fi
}
```

\PdfRender@PatternFillColor

```
\edef\PdfRender@PatternFillColor{ % space
  (%
    [0-9\string\.]+ g|%
    [0-9\string\.]+ [0-9\string\.]+ [0-9\string\.]+ rg|%
    [0-9\string\.]+ [0-9\string\.]+ %
    [0-9\string\.]+ [0-9\string\.]+ k%
  ) % space
  (.*)$%
}
```

\PdfRender@PatternStrokeColor

```
\edef\PdfRender@PatternStrokeColor{ % space
  (%
    [0-9\string\.]+ G|%
    [0-9\string\.]+ [0-9\string\.]+ [0-9\string\.]+ RG|%
    [0-9\string\.]+ [0-9\string\.]+ %
    [0-9\string\.]+ [0-9\string\.]+ K%
  ) % space
  (.*)$%
}
```

\PdfRender@MatchPattern

```
\def\PdfRender@MatchPattern#1{%
  \ifnum\pdfmatch{\PdfRender@Pattern}{\PdfRender@String}=1 %
    \xdef#1{%
      \expandafter\strip@prefix\pdflastmatch 1%
    }%
    \edef\PdfRender@String{%
      \expandafter\strip@prefix\pdflastmatch 2%
    }%
    \ifx\PdfRender@String\ltx@empty
    \else
      \expandafter\expandafter\expandafter\PdfRender@MatchPattern
      \expandafter\expandafter\expandafter#1%
    \fi
  \fi
}
```

\PdfRender@GetFillColor

```
\def\PdfRender@GetFillColor{%
  \global\let\PdfRender@FillColor\ltx@empty
  \begingroup
    \ifPdfRender@Match
```

```
    \let\PdfRender@Pattern\PdfRender@PatternFillColor
    \edef\PdfRender@String{\ltx@space\current@color\ltx@space}%
    \PdfRender@MatchPattern\PdfRender@FillColor
   \else
    \edef\current@color{\current@color\ltx@space}%
    \let\PdfRender@OP\relax
    \PdfRender@FindOp{g}0%
    \PdfRender@FindOp{G}1%
    \PdfRender@FindOp{rg}0%
    \PdfRender@FindOp{RG}1%
    \PdfRender@FindOp{k}0%
    \PdfRender@FindOp{K}1%
    \PdfRender@FilterOp 0\PdfRender@FillColor
   \fi
  \endgroup
}
```

\PdfRender@GetStrokeColor

```
\def\PdfRender@GetStrokeColor{%
  \global\let\PdfRender@StrokeColor\ltx@empty
  \begingroup
   \ifPdfRender@Match
    \let\PdfRender@Pattern\PdfRender@PatternStrokeColor
    \edef\PdfRender@String{\ltx@space\current@color\ltx@space}%
    \PdfRender@MatchPattern\PdfRender@StrokeColor
   \else
    \edef\current@color{\current@color\ltx@space}%
    \let\PdfRender@OP\relax
    \PdfRender@FindOp{g}0%
    \PdfRender@FindOp{G}1%
    \PdfRender@FindOp{rg}0%
    \PdfRender@FindOp{RG}1%
    \PdfRender@FindOp{k}0%
    \PdfRender@FindOp{K}1%
    \PdfRender@FilterOp 1\PdfRender@StrokeColor
   \fi
  \endgroup
}

\ifPdfRender@Match
  \expandafter\PdfRender@AtEnd
\fi%
```

\PdfRender@FindOp

```
\def\PdfRender@FindOp#1#2{%
  \def\PdfRender@temp##1 #1 ##2\@nil{%
    ##1%
    \ifx\\##2\\%
      \expandafter\@gobble
    \else
      \PdfRender@OP{#1}#2%
      \expandafter\@firstofone
    \fi
    {%
      \PdfRender@temp##2\@nil
    }%
  }%
  \edef\current@color{%
    \@firstofone{\expandafter\PdfRender@temp\current@color} #1 \@nil
  }%
}
```

\PdfRender@FilterOp

```
\def\PdfRender@FilterOp#1#2{%
  \expandafter\PdfRender@@FilterOp\expandafter#1\expandafter#2%
    \current@color\PdfRender@OP{}{}%
}
```

\PdfRender@@FilterOp

```
\def\PdfRender@@FilterOp#1#2#3\PdfRender@OP#4#5{%
  \ifx\\#4#5\\%
  \else
    \ifnum#1=#5 %
      \xdef#2{#3 #4}%
    \fi
    \expandafter\PdfRender@@FilterOp\expandafter#1\expandafter#2%
  \fi
}

\PdfRender@AtEnd%
⟨/package⟩
```

3 Test

3.1 Catcode checks for loading

```
⟨*test1⟩
\catcode`\{=1 %
\catcode`\}=2 %
\catcode`\#=6 %
\catcode`\@=11 %
\expandafter\ifx\csname count@\endcsname\relax
  \countdef\count@=255 %
\fi
\expandafter\ifx\csname @gobble\endcsname\relax
  \long\def\@gobble#1{}%
\fi
\expandafter\ifx\csname @firstofone\endcsname\relax
  \long\def\@firstofone#1{#1}%
\fi
\expandafter\ifx\csname loop\endcsname\relax
  \expandafter\@firstofone
\else
  \expandafter\@gobble
\fi
{%
  \def\loop#1\repeat{%
    \def\body{#1}%
    \iterate
  }%
  \def\iterate{%
    \body
      \let\next\iterate
    \else
      \let\next\relax
    \fi
    \next
  }%
  \let\repeat=\fi
}%
\def\RestoreCatcodes{}
\count@=0 %
\loop
  \edef\RestoreCatcodes{%
    \RestoreCatcodes
```

```
    \catcode\the\count@=\the\catcode\count@\relax
  }%
\ifnum\count@<255 %
  \advance\count@ 1 %
\repeat

\def\RangeCatcodeInvalid#1#2{%
  \count@=#1\relax
  \loop
    \catcode\count@=15 %
  \ifnum\count@<#2\relax
    \advance\count@ 1 %
  \repeat
}
\def\RangeCatcodeCheck#1#2#3{%
  \count@=#1\relax
  \loop
    \ifnum#3=\catcode\count@
    \else
      \errmessage{%
        Character \the\count@\space
        with wrong catcode \the\catcode\count@\space
        instead of \number#3%
      }%
    \fi
  \ifnum\count@<#2\relax
    \advance\count@ 1 %
  \repeat
}
\def\space{ }
\expandafter\ifx\csname LoadCommand\endcsname\relax
  \def\LoadCommand{\input pdfrender.sty\relax}%
\fi
\def\Test{%
  \RangeCatcodeInvalid{0}{47}%
  \RangeCatcodeInvalid{58}{64}%
  \RangeCatcodeInvalid{91}{96}%
  \RangeCatcodeInvalid{123}{255}%
  \catcode`\@=12 %
  \catcode`\\=0 %
  \catcode`\%=14 %
  \LoadCommand
  \RangeCatcodeCheck{0}{36}{15}%
  \RangeCatcodeCheck{37}{37}{14}%
  \RangeCatcodeCheck{38}{47}{15}%
  \RangeCatcodeCheck{48}{57}{12}%
  \RangeCatcodeCheck{58}{63}{15}%
  \RangeCatcodeCheck{64}{64}{12}%
  \RangeCatcodeCheck{65}{90}{11}%
  \RangeCatcodeCheck{91}{91}{15}%
  \RangeCatcodeCheck{92}{92}{0}%
  \RangeCatcodeCheck{93}{96}{15}%
  \RangeCatcodeCheck{97}{122}{11}%
  \RangeCatcodeCheck{123}{255}{15}%
  \RestoreCatcodes
}
\Test
\csname @@end\endcsname
\end
```

⟨/test1⟩

3.2 Simple test file

```
⟨*test2⟩
\NeedsTeXFormat{LaTeX2e}
\ProvidesFile{pdfrender-test2.tex}[2016/05/17]
\documentclass{article}
\usepackage{color}
\usepackage{pdfrender}[2016/05/17]
\begin{document}
Hello World
\newpage
Start
\textpdfrender{%
  TextRenderingMode=1,%
  LineWidth=.1,%
  LineCapStyle=2,%
  LineJoinStyle=1,%
  MiterLimit=1.2,%
  LineDashPattern=[2 2]0,%
  RenderingIntent=Saturation,%
}{Hello\newpage World}
Stop
\par
\newlength{\LineWidth}
\setlength{\LineWidth}{.5pt}
Start
\textpdfrender{%
  FillColor=yellow,%
  StrokeColor=[cmyk]{1,.5,0,0},%
  TextRenderingMode=FillStroke,%
  LineWidth=.5\LineWidth,%
  LineCapStyle=Round,%
  LineJoinStyle=Bevel,%
}{Out-\par\newpage line}
Stop
\end{document}
⟨/test2⟩
```

3.3 Further tests

Robustness and bookmarks.

```
⟨*test3⟩
\NeedsTeXFormat{LaTeX2e}
\ProvidesFile{pdfrender-test3.tex}[2016/05/17]
\documentclass{article}
\usepackage{pdfrender}[2016/05/17]
\usepackage{hyperref}
\usepackage{bookmark}
\begin{document}
\tableofcontents
\section{%
  \textpdfrender{%
    TextRenderingMode=1,%
    LineCapStyle=2,%
    LineJoinStyle=1,%
    MiterLimit=1.2,%
    LineDashPattern=[2 2]0,%
    RenderingIntent=Saturation,%
  }{Hello World}%
}
\end{document}
⟨/test3⟩
```

Color algorithm if `\pdfmatch` is not available.

```
⟨*test4⟩
```

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesFile{pdfrender-test4.tex}[2016/05/17]
\documentclass[12pt]{article}
\usepackage{pdfrender}[2016/05/17]
\usepackage{color}
\usepackage{qstest}
\IncludeTests{*}
\LogTests{log}{*}{*}
\makeatletter
\newcommand*{\CheckColor}[1]{%
  \Expect{#1}*{\current@color}%
}
\makeatother
\begin{document}
  \begin{qstest}{color}{color}%
    \CheckColor{0 g 0 G}%
    \Huge\bfseries
    \noindent
    \textpdfrender{%
      TextRenderingMode=2,%
      LineWidth=.5,%
      FillColor=yellow,%
      StrokeColor=blue,%
    }{%
      \CheckColor{0 0 1 0 k 0 0 1 RG}%
      Blue(Yellow)\\%
      \textpdfrender{%
        FillColor=green,%
      }{%
        \CheckColor{0 1 0 rg 0 0 1 RG}%
        Blue(Green)%
      }\\%
      \CheckColor{0 0 1 0 k 0 0 1 RG}%
      Blue(Yellow)\\%
      \textpdfrender{%
        StrokeColor=red,%
      }{%
        \CheckColor{0 0 1 0 k 1 0 0 RG}%
        Red(Yellow)%
      }\\%
      \CheckColor{0 0 1 0 k 0 0 1 RG}%
      Blue(Yellow) %
    }%
  \end{qstest}%
  \begin{qstest}{colorlast}{colorlast}%
    \makeatletter
    \def\Test#1#2#3{%
      \begingroup
        \def\current@color{#1}%
        \textpdfrender{#2}{%
          \CheckColor{#3}%
        }%
      \endgroup
    }%
    \Test{1 g 0 0 1 RG 0 0 1 0 k 0.5 G}%
         {StrokeColor=green}%
         {0 0 1 0 k 0 1 0 RG}%
    \Test{1 g 0 0 1 RG 0 0 1 0 k 0.5 G}%
         {FillColor=red}%
         {1 0 0 rg 0.5 G}%
  \end{qstest}%
\end{document}
```

```
⟨/test4⟩
```

3.4 Compatibility with plain TEX

```
⟨*test5⟩
\input luatex85.sty
\pdfoutput=1 %
\hsize=6.5in
\vsize=8.9in
\pdfpagewidth=\hsize
\pdfpageheight=\vsize
\parfillskip=0pt plus 1fil\relax
\input pdfrender.sty\relax
\catcode`\{=1 %
\catcode`\}=2 %
\let\OrgMakeFootLine\makefootline
\def\makefootline{%
  \begingroup\normalcolor\OrgMakeFootLine\endgroup
}
\font\f=ec-lmr10 scaled 3000\relax
\f
Before %
\textpdfrender{%
  TextRenderingMode=1,%
  LineWidth=.1,%
}{Hello\par\vfill\penalty-10000 World} %
After %
\par
\vfill
\penalty-10000 %
\csname @@end\endcsname\end
⟨/test5⟩
```

4 Installation

4.1 Download

Package. This package is available on CTAN[1]:

CTAN:macros/latex/contrib/oberdiek/pdfrender.dtx The source file.

CTAN:macros/latex/contrib/oberdiek/pdfrender.pdf Documentation.

Bundle. All the packages of the bundle 'oberdiek' are also available in a TDS compliant ZIP archive. There the packages are already unpacked and the documentation files are generated. The files and directories obey the TDS standard.

CTAN:install/macros/latex/contrib/oberdiek.tds.zip

TDS refers to the standard "A Directory Structure for TEX Files" (CTAN:tds/tds.pdf). Directories with `texmf` in their name are usually organized this way.

4.2 Bundle installation

Unpacking. Unpack the `oberdiek.tds.zip` in the TDS tree (also known as `texmf` tree) of your choice. Example (linux):

```
unzip oberdiek.tds.zip -d ~/texmf
```

[1] http://ctan.org/pkg/pdfrender

Script installation. Check the directory TDS:scripts/oberdiek/ for scripts that need further installation steps. Package attachfile2 comes with the Perl script pdfatfi.pl that should be installed in such a way that it can be called as pdfatfi. Example (linux):

```
chmod +x scripts/oberdiek/pdfatfi.pl
cp scripts/oberdiek/pdfatfi.pl /usr/local/bin/
```

4.3 Package installation

Unpacking. The .dtx file is a self-extracting docstrip archive. The files are extracted by running the .dtx through plain TEX:

```
tex pdfrender.dtx
```

TDS. Now the different files must be moved into the different directories in your installation TDS tree (also known as texmf tree):

pdfrender.sty	→ tex/generic/oberdiek/pdfrender.sty
pdfrender.pdf	→ doc/latex/oberdiek/pdfrender.pdf
test/pdfrender-test1.tex	→ doc/latex/oberdiek/test/pdfrender-test1.tex
test/pdfrender-test2.tex	→ doc/latex/oberdiek/test/pdfrender-test2.tex
test/pdfrender-test3.tex	→ doc/latex/oberdiek/test/pdfrender-test3.tex
test/pdfrender-test4.tex	→ doc/latex/oberdiek/test/pdfrender-test4.tex
test/pdfrender-test5.tex	→ doc/latex/oberdiek/test/pdfrender-test5.tex
pdfrender.dtx	→ source/latex/oberdiek/pdfrender.dtx

If you have a docstrip.cfg that configures and enables docstrip's TDS installing feature, then some files can already be in the right place, see the documentation of docstrip.

4.4 Refresh file name databases

If your TEX distribution (teTEX, mikTEX, ...) relies on file name databases, you must refresh these. For example, teTEX users run texhash or mktexlsr.

4.5 Some details for the interested

Unpacking with LATEX. The .dtx chooses its action depending on the format:

plain TEX: Run docstrip and extract the files.

LATEX: Generate the documentation.

If you insist on using LATEX for docstrip (really, docstrip does not need LATEX), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{pdfrender.dtx}
```

Do not forget to quote the argument according to the demands of your shell.

Generating the documentation. You can use both the .dtx or the .drv to generate the documentation. The process can be configured by the configuration file ltxdoc.cfg. For instance, put this line into this file, if you want to have A4 as paper format:

```
\PassOptionsToClass{a4paper}{article}
```

An example follows how to generate the documentation with pdfLATEX:

```
pdflatex pdfrender.dtx
makeindex -s gind.ist pdfrender.idx
pdflatex pdfrender.dtx
makeindex -s gind.ist pdfrender.idx
pdflatex pdfrender.dtx
```

5 Catalogue

The following XML file can be used as source for the TEX Catalogue. The elements caption and description are imported from the original XML file from the Catalogue. The name of the XML file in the Catalogue is pdfrender.xml.

```
⟨*catalogue⟩
<?xml version='1.0' encoding='us-ascii'?>
<!DOCTYPE entry SYSTEM 'catalogue.dtd'>
<entry datestamp='$Date$' modifier='$Author$' id='pdfrender'>
  <name>pdfrender</name>
  <caption>Control rendering parameters.</caption>
  <authorref id='auth:oberdiek'/>
  <copyright owner='Heiko Oberdiek' year='2010'/>
  <license type='lppl1.3'/>
  <version number='1.4'/>
  <description>
    The package provides interfaces for the user to control PDF
    parameters, such as line width or text rendering mode.  The
    control operations work in a manner very similar to that of the
     <xref refid='color'>color</xref> package.
    <p/>
    The package is part of the <xref refid='oberdiek'>oberdiek</xref> bundle.
  </description>
  <documentation details='Package documentation'
      href='ctan:/macros/latex/contrib/oberdiek/pdfrender.pdf'/>
  <ctan file='true' path='/macros/latex/contrib/oberdiek/pdfrender.dtx'/>
  <miktex location='oberdiek'/>
  <texlive location='oberdiek'/>
  <install path='/macros/latex/contrib/oberdiek/oberdiek.tds.zip'/>
</entry>
⟨/catalogue⟩
```

6 Acknowledgement

Friedrich Vosberg asked in the newsgroup de.comp.text.tex for the font outline feature [2].

Gaius Pupus proposed the basic method using \pdfliteral in this thread [3].

Rolf Niepraschk added color support [4].

7 References

[1] Adobe Systems Incorporated. *PDF Reference – Adobe Portable Document format – Version 1.7*. 6th ed. 2006. URL: http://www.adobe.com/devnet/acrobat/pdfs/pdf_reference_1-7.pdf.

[2] Friedrich Vosberg, *Text in Buchstabenumrissen*, de.comp.text.tex, 2010-01-22. URL: http://groups.google.com/group/de.comp.text.tex/msg/f442310ac8b2d506.

[3] Gaius Pupus, *Re: Text in Buchstabenumrissen*, de.comp.text.tex, 2010-01-23. URL: http://groups.google.com/group/de.comp.text.tex/msg/95d890d77ac47eb1.

[4] Rolf Niepraschk, *Re: Text in Buchstabenumrissen*, de.comp.text.tex, 2010-01-24. URL: http://groups.google.com/group/de.comp.text.tex/msg/4eb61a5879db54db.

8 History

[2010/01/26 v1.0]

- The first version.

[2010/01/27 v1.1]

- Macros \pdfrender and \textpdfrender are made robust.
- Color extraction rewritten for the case that \pdfmatch is not available. This fixes wrong color assigments in case of nesting.
- Color extraction of case \pdfmatch is fixed for the case that the color string contains several fill or several stroke operations.

[2010/01/28 v1.2]

- Dependency from package color is removed.
- Compatibility for plain TEX and even iniTEX added.

[2016/05/14 v1.3]

- Use package luatex85 for compatibility with new LuaTeX.

[2016/05/17 v1.4]

- Documentation updates.
- adjust luatex85 reference so that it works in plain TeX.

9 Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; plain numbers refer to the code lines where the entry is used.

Symbols

\# 758
\% 834
\. . 481, 520, 521, 524, 528, 530, 649, 650, 651, 652, 658, 659, 660, 661
\@ 759, 832
\@PackageError 584
\@PackageInfo 333
\@PackageInfoNoLine 178
\@PackageWarning 171, 187, 551, 572, 626, 637
\@PackageWarningNoLine 149
\@ehc 431, 587
\@empty 447
\@firstofone 730, 737, 767, 770
\@gobble 727, 764, 772
\@ifnextchar 595
\@ifundefined 583
\@nil 724, 733, 737
\@nodocument 236, 254
\@undefined 58, 163
\[. 527
\\ 497, 726, 745, 833, 936, 942, 944, 950
\{ 756, 983
\} 757, 984
\] 529

A

\advance 797, 805, 820
\afterassignment 490
\AfterEndPreamble 271, 276
\aftergroup 29, 346, 360
\AfterPackage 268, 269, 270
\AtBeginDocument 264

B

\begin 860, 896, 924, 925, 955
\bfseries 927
\body 776, 780

C

\catcode 2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 33, 34, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 69, 70, 72, 73, 74, 78, 79, 80, 81, 82, 83, 84, 87, 88, 90, 91, 92, 93, 97, 99, 756, 757, 758, 759, 794, 803, 811, 815, 832, 833, 834, 983, 984
\chardef 327, 601, 608, 615
\CheckColor 920, 926, 935, 940, 943, 948, 951, 961
\color 595, 598
\color@begingroup 201, 203, 204
\color@endbox 205, 216
\color@endgroup 202, 205, 245
\color@hbox 203, 216
\color@setgroup 200, 245, 248
\color@vbox 204
\count@ 761, 790, 794, 796, 797, 801, 803, 804, 805, 809, 811, 814, 815, 819, 820
\countdef 761
\csname 14, 21, 50, 66, 76, 129, 131, 134, 136, 139, 144, 153, 322, 323, 324, 326, 328, 331, 335, 337, 339, 340, 342, 344, 347, 349, 350, 352, 356, 358, 361, 363, 365, 369, 371, 372, 377, 384, 387, 392, 399, 400, 403, 411, 417, 418, 422, 423, 430, 432, 436, 437, 445, 450, 456, 460, 461, 463, 760, 763, 766, 769, 824, 851, 1000
\current@color 617, 630, 641, 685, 688, 705, 708, 736, 737, 742, 921, 959

D

\DeclareRobustCommand 288, 305
\define@key 414
\dimen 491, 500, 503
\dimexpr 486
\documentclass 857, 892, 913

E

\empty 17, 18
\end . 852, 887, 908, 954, 971, 972, 1000
\endcsname 14, 21, 50, 66, 76, 129, 131, 134, 136, 139, 144, 153, 322, 323, 324, 326, 328, 331, 335, 337, 339, 340, 342, 344, 347, 349, 350, 352, 356, 358, 361, 363, 365, 369, 371, 372, 377, 384, 387, 392, 399, 400, 403, 411, 417, 418, 422, 423, 430, 432, 436, 437, 445, 450, 456, 460, 461, 463, 760, 763, 766, 769, 824, 851, 1000
\endgraf 202, 210
\endinput 29, 126
\endlinechar 4, 35, 71, 77, 89
\errmessage 813
\Expect 921

F

\f 989, 990
\font 989

G

\gdef 220, 337

H

\hbox 203
\hsize 977, 979
\Huge 927

I

\iffalse 137
\ifnum 436, 623, 634, 666, 747, 796, 804, 811, 819
\ifpdf 169
\ifPdfRender@Match 142, 434, 683, 703, 720
\ifPdfRender@Stack 141, 325
\ifPdfRender@Values 315, 382
\iftrue 132
\ifx 15, 18, 21, 50, 58, 61, 144, 163, 227, 236, 246, 254, 385, 447, 497, 543, 550, 564, 571, 602, 603, 611, 625, 636, 673, 726, 745, 760, 763, 766, 769, 824
\immediate 23, 52
\IncludeTests 917
\input 158, 825, 975, 982
\iterate 777, 779, 781

K

\kv@key 384, 387
\kv@parse@normalized 383
\kv@value 385, 387
\kvsetkeys 295

L

\LineWidth 875, 876, 882
\LoadCommand 825, 835
\LogTests 918
\loop 775, 791, 802, 810
\ltx@empty 293, 294, 432, 543, 544, 548, 550, 564, 565, 569, 571, 597, 602, 603, 611, 625, 636, 673, 681, 701
\ltx@firstofone 590
\ltx@firstoftwo 280, 283, 300
\ltx@GlobalAppendToMacro 229, 248, 375, 379
\ltx@gobble 588
\ltx@gobbletwo 389
\ltx@ifpackagelater 148
\ltx@ifpackageloaded 199
\ltx@IfUndefined 170, 177, 190, 263, 266, 274, 280, 282, 299, 401, 409, 455, 483

\ltx@ifundefined 206, 209, 219
\ltx@pkgextension 153
\ltx@secondoftwo 283, 300
\ltx@space 179, 466, 467, 606, 613, 618, 685, 688, 705, 708

M

\makeatletter 919, 956
\makeatother 923
\makefootline 985, 986
\MessageBreak 150, 151, 152, 153, 180, 428, 552, 573, 585

N

\NeedsTeXFormat 855, 890, 911
\newcommand . . 194, 195, 287, 304, 920
\newlength 875
\newpage 862, 872, 885
\next 781, 783, 785
\noindent 928
\normalcolor 220, 226, 229, 987
\number 334, 816

O

\OrgMakeFootLine 985, 987

P

\PackageError 427
\PackageInfo 26
\par 210, 874, 885, 995, 997
\parfillskip 981
\pdfcolorstack 341, 351
\pdfcolorstackinit 172, 329
\pdfextension 163
\pdflastmatch 668, 671
\pdfliteral 357, 364
\pdfmatch 179, 436, 666
\pdfoutput 976
\pdfpageheight 980
\pdfpagewidth 979
\pdfrender *2*, 191, 194, 282, 310
\PdfRender@@FilterOp 741, 744
\PdfRender@@PostProcessLineWidth 490, 496
\PdfRender@@TryColor 595, 597
\PdfRender@AtEnd 95, 96, 123, 125, 197, 721, 753
\PdfRender@AtEndHook 122, 124, 146, 147
\PdfRender@Color 542, 543, 548, 563, 564, 569, 598
\PdfRender@ColorAvailable 546, 567, 582
\PdfRender@ColorSetGroupHook 214, 242, 249, 379
\PdfRender@Current 393, 444, 447, 455, 457, 461
\PdfRender@CurrentColor 605, 612, 630, 641
\PdfRender@CurrentLineWidth 491, 502
\PdfRender@define@key 391, 398, 409, 540, 561
\PdfRender@dimexpr 484, 486, 491
\PdfRender@ErrorInvalidValue 426, 439, 448, 458
\PdfRender@FillColor 293, 544, 550, 602, 606, 613, 618, 625, 681, 686, 696
\PdfRender@FilterOp . . . 696, 716, 740
\PdfRender@FindOp 690, 691, 692, 693, 694, 695, 710, 711, 712, 713, 714, 715, 723
\PdfRender@GetFillColor 549, 624, 680
\PdfRender@GetStrokeColor 570, 635, 700
\PdfRender@let 492, 494, 496
\PdfRender@MatchPattern 665, 686, 706
\PdfRender@Matchtrue 184
\PdfRender@NeedsCurrentColor 601, 608, 615, 623, 634
\PdfRender@NewClass 318, 320, 480, 519, 523, 526
\PdfRender@NewClassValues 316, 468, 509, 514, 532
\PdfRender@newif 128, 141, 142, 315, 321
\PdfRender@NormalColorHook 213, 220, 223, 230, 375
\PdfRender@OP 689, 709, 729, 742, 744
\PdfRender@OpName 467, 534
\PdfRender@OpValue 466, 470, 480, 509, 514, 519, 523, 526
\PdfRender@PatchColor 258, 262, 264, 268, 269, 271, 276
\PdfRender@PatchColorSetGroup 240, 260
\PdfRender@PatchNormalColor 218, 259, 292
\PdfRender@Pattern 666, 684, 704
\PdfRender@PatternFillColor . 647, 684
\PdfRender@PatternStrokeColor 656, 704
\PdfRender@PostProcessLineWidth 488
\PdfRender@relax 492, 495
\PdfRender@RequirePackage 143, 164, 166, 167, 168, 278
\PdfRender@Reset 421
\PdfRender@Set 373, 380, 416
\PdfRender@SetColor 296, 600
\PdfRender@SetValidate 405, 434
\PdfRender@SetValidateValues 394, 454
\PdfRender@Stacktrue 175
\PdfRender@String 666, 670, 673, 685, 705
\PdfRender@StrokeColor 294, 565, 571, 603, 606, 611, 613, 618, 636, 701, 706, 716
\PdfRender@temp 223, 227, 242, 246, 724, 733, 737
\PdfRender@TestBox 215, 224, 243, 547, 568
\PdfRender@texorpdfstring 279, 291, 308
\PdfRender@TryColor . . . 548, 569, 594
\PdfRender@Valuesfalse 396
\PdfRender@Valuestrue 317
\penalty 995, 999
\ProvidesFile 856, 891, 912

\ProvidesPackage 19, 67

R

\RangeCatcodeCheck 808, 836, 837, 838, 839, 840, 841, 842, 843, 844, 845, 846, 847
\RangeCatcodeInvalid 800, 828, 829, 830, 831
\repeat 775, 787, 798, 806, 821
\RequirePackage 161
\RestoreCatcodes . . 789, 792, 793, 848

S

\section 898
\set@color . 200, 225, 244, 620, 631, 642
\setbox 216
\setlength 876
\space 814, 815, 823
\strip@prefix 668, 671
\strip@pt 503

T

\tableofcontents 897
\Test 827, 850, 957, 965, 968
\texorpdfstring 280
\textpdfrender *2*, 192, 195, 299, 864, 878, 899, 929, 937, 945, 960, 992
\the 77, 78, 79, 80, 81, 82, 83, 84, 97, 794, 814, 815
\TMP@EnsureCode . . . 94, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121

U

\usepackage 858, 859, 893, 894, 895, 914, 915, 916

V

\vbox 204
\vfill 995, 998
\vsize 978, 980

W

\write 23, 52

X

\x 14, 15, 18, 22, 26, 28, 51, 56, 66, 75, 87, 501, 506