PDCFSEL, a font-selection scheme for TEX

Version 3.5, last changed pdc 1995–03–28

P. Damian Cugley

1. Introduction

This document describes PDCFSEL, a file of font selection macros designed to be used with documents using the plain TEX format (rather than LaTEX, for example).

The PDCFSEL macros perform a similar function to the so-called New Font Selection Scheme 2 (used in LaTEX 2e). PDCFSEL uses a simpler model of font selection, and so is a smaller package—about 110 lines of code—which is just as well as a copy of it will probably need to be included with documents using it. The description of which fonts are used in a document can be relatively compact, despite not using any special database files.

1.1. *Organizing fonts into fontsets*

We can arrange the fonts used in an imaginary TEX document in a table like so (with an asterisk marking fonts that have to be scaled to fit):

	`\rm`	`\it`	`\bf`	`\bi`	`\mi`	`\sy`
body text	`cmr10`	`cmti10`	`cmbx10`	`cmbxti10`	`cmmi10`	`cmsy10`
footnotes	`cmr8`	`cmti8`	`cmbx8`	`cmbxti10`*	`cmmi8`	`cmsy8`
script	`cmr7`	`cmti7`	`cmbx7`	`cmbxti10`*	`cmmi7`	`cmsy7`
scriptscript	`cmr5`	`cmti7`*	`cmbx5`	`cmbxti10`*	`cmmi5`	`cmsy5`
heading	`cmss12`	`cmssi12`	-	-	-	-
subheading	`cmss10`	`cmssi10`	-	-	-	-

The column headings are *font nicknames*. These nicknames are used as described in the *TEXbook* to switch between fonts in the same row. Each has a corresponding *fam* used in maths mode, with symbolic names like `\itfam` and `\bffam` (we define `\rmfam` $=$ 0 for consistency). I have added nicknames `\mi` and `\sy` for maths italic and maths symbol respectively.

Each row of the table is a *fontset*. Plain TEX defines a single fontset (which is like the 'body text' row of this table); in Appendix E, Knuth discusses formats that switch between different fontsets with macros like `\ninepoint` and `\eightpoint`. LaTEX 2.09 uses many fontsets, called `\normalsize`, `\large`, and so on. (NFSS 2 uses a more complex system, where size, weight, and slant may be changed independently of each other.)

In this table, subheadings and body text have different fontsets in the above, even though the fonts are the same size. Also, there are no `\sf` fonts; the headings are printed in sanserif by using a fontset with `\rm` mapped on to a sanserif font. This is more logical and flexible than the LaTEX approach.

We can call font families which allow this table to work tidily 'regular', and families (like Computer Modern) that require exceptions 'irregular'. The above table has one exception (`\it` in scriptscript must use a scaled font); the CM Bold Extended Text Italic fonts can be described as regular if we think of the 'body text' entry as being '`cmbxti10 at 10pt`', so that the fonts in that column are scaled fonts without exception. To make font specification compact, we want to take advantage of regular families as much as possible, while not making it difficult to include exceptions.

1.2. *Introduction to using PCDFSEL*

PDCFSEL uses no databases of font families, and loads no fonts by default. This makes it more flexible, but also requires document designers to write a 'mini-database' of those fonts used in the document. The above scheme might be defined as follows:

```
\input pdcfsel

\newfam\bifam

\def\texttemplate{%
    \m{rm}{cmr}\m{it}{cmti}\m{bf}{cmbx}\@\m{bi}{cmbxti10}%
    \M{mi}{cmmi}\M{sy}{cmsy}%
}
\loadfont\scriptscriptit{cmti7 at 5pt }
\xfontset{scriptscript}\texttemplate{5}
\xfontset{script}\texttemplate{7}
\fontset{note}\texttemplate{8}{10pt}{scriptscript}{scriptscript}
\fontset{body}\texttemplate{10}{12pt}{script}{scriptscript}

\def\headingtemplate{%
    \f{rm}{cmss}\f{it}{cmssi}%
}
\fontset{subheading}\headingtemplate{10}{12pt}{subheading}{subheading}
\fontset{heading}\headingtemplate{12}{14pt}{subheading}{subheading}

\bodyfonts
```

This defines commands `\bodyfonts`, `\notefonts`, `\headingfonts` and `\subheadingfonts` which switch between fontsets. (The *script* and *scriptscript* fontsets, which are defined with `\xfontset`, are used only in maths mode and don't need '`-fonts`' commands.) These `-fonts` commands are not usually used directly in documents; `\notefonts` will be used in some `\footnote` command, `\headingfonts` in some heading-generating command, and so on.

Changes of fontset are accompanied by changes in parameters like `\baselineskip` and pseudo-parameters like `\smallskipamount`, and give definitions to the *maths font tables* `\textfont\rmfam`, . . . , `\scriptscriptfont\syfam` (for those fonts that will be used in maths mode).

We want font nickname commands to be efficient, because they are expected to be more common than changes between rows in the table. In this implementation, after an invocation of `\bodyfonts`, the macro `\rm` expands to exactly '`\fam\rmfam \bodyrm`'.

1.3. *How the rest of this document is organized*

The remainder of this document is a description of all of `pdcfsel.tex`, including more details of how the commands it defines are used.

Running this document (`pdcfsel.dtx`) through plain TEX creates the definitions file (`pdcfsel.tex`) in addition to the usual `dvi` and `log` files. This way the macros and their documentation may be kept together in one file. The definition lines are numbered.

The definitions start with macros for loading individual fonts, followed by the macros used to group fonts into fontsets.

2. Getting started

2.1. *File identification*

We start with some comments indentifying the file.

```
% pdcfsel.tex -- macros for loading fonts -*-tex-*-

%%%@TeX-document-file {
%%% title      = "PDC Font Selection Scheme",
%%% filename   = "$texmf/tex/plain/pdcmac/pdcfsel.tex",
%%% version    = "3.5",
%%% Date       = "1995/03/28",
%%% creator    = "pdcfsel.dtx",
%%% package    = "pdcmac 1.0",
%%% author     = "P. Damian Cugley",
%%% email      = "damian.cugley@comlab.ox.ac.uk",
%%% address    = "Oxford University Computing Laboratory,"
%%%               Parks Road, Oxford  OX1 3QD, UK",
%%% abstract   = "A file of definitions for managing font
%%%               selection in documents based on the plain
%%%               TeX macros.
%%%                 This file was generated by running
%%%               plain TeX on pdcfsel.dtx.",
%%% dependencies = ""
%%%}

\message{3.5 <pdc 1995/03/28>}
```

2.2. *Private names*

Macros internal to FSEL all have names starting '`\FSEL`'.

2.3. *Macros for edefs*

These save typing `\expandafter` a lot. The expression '`\expcs`⟨*token*⟩`{`⟨*tokens*⟩`}`', creates a csname from ⟨*tokens*⟩ and applies ⟨*token*⟩ to the result. In the body of an `\edef`, The expression '`\noexpcs{`⟨*tokens*⟩`}`' converts ⟨*tokens*⟩ to a csname without expanding the result.

```
\def\expcs#1#2{\expandafter#1\csname#2\endcsname}
\def\noexpcs{\expcs\noexpand}
```

3. Selecting auto-loading mode

The flag `\ifFSELautoload` is true, fonts are auto-loaded when they first used instead of all at the start. This is useful when not all the fonts described by the fontsets will be needed. The flag is set with the user command `\autoloadfonts`.

When demand-loading, the fonts used are written to a file named after the document with a `.fnt` suffix. This checklist might be used to decide which fonts need to be sent with the document if it is being sent to someone to compile on a different TEX system.

```
\newif\ifFSELautoload
\def\autoloadfonts{
    \FSELautoloadtrue
    \csname newwrite\endcsname \FSELfile
    \immediate\openout\FSELfile=\jobname.fnt
}
```

4. How to set up the csname for one font

The macro `\loadfont` is used to load individual fonts, defining a control sequence name (csname) which may be used later to switch to that font. When not demand-loading fonts, this is just like `\global\font` followed by expanding the `\everyloadfont` macro.

If the csname is already defined, then this command does nothing. This is so that irregularities in the font scheme for the document can be allowed for. Parameter `#1` is the csname, for example '`\bodyrm`', and `#2` is the external name, for example '`cmr10`' or '`cmr10 at 12pt`'.

```
\def\loadfont#1#2{%
    \ifx#1\relax
        \FSELloadfont#1{#2}%
    \else\ifx#1\UNDEFINED
        \FSELloadfont#1{#2}%
    \fi\fi
}
```

(We have to compare `#1` against both `\relax`, which is produced by `\csname`–`\endcsname`, and a completely undefined csname.) The macro `\FSELloadfont` doesa the actual work of loading the font.

```
\def\FSELloadfont#1#2{%
    \ifFSELautoload
```

Demand-loading. We don't load the font, instead we define a csname as a macro. When expanded this new macro will (a) write the font name to the `fnt` file; (b) define the macro `\subfont` to load `cmr10` instead (in case TeX stops with a 'font not loadable' message); (c) load the font for real (this overwrites the macro); (d) call `\everyloadfont` for per-font customization and (e) switch to the new font.

```
        \edef#1{%
            \write\FSELfile{#2}%
            \def\noexpand\subfont{\global\font\noexpand#1cmr10 }%
            \global\font\noexpand#1#2\relax
            \noexpand\everyloadfont\noexpand#1{#2}%
            \noexpand#1%
        }%
    \else
```

Immediate loading.

```
        \global\font#1#2\relax  \everyloadfont#1{#2}%
    \fi
}
```

The macro `\everyloadfont` is expanded immediately after actually loading a font. (The definition used is that one current when the font is actually loaded, not the one current when `\loadfont` was executed.) Its `#1` parameter is always a ⟨*fontdef token*⟩, i.e., it can be to used as a parameter to `\fontdimen1`. The `#2` parameter is the external name of the font.

```
\def\everyloadfont#1#2{}
```

For example, in a document with a ragged-right margin, this might be used to suppress the stretch and shrink of interword spaces by being defined as follows

```
\def\everyloadfont#1#2{%
    \fontdimen3#1=0pt \fontdimen4#1=0pt
}
```

5. How to set up a fontset

Now that we know how to define individual font csnames, we need the mechanism for grouping them in to fontsets. A fontset is defined by a template macro which says what font nicknames are defined and gives part of the external font name. The template macro takes no parameters and expands to a list each of whose elements are of the form

⟨*type*⟩`{`⟨*nickname*⟩`}{`⟨*partial name*⟩`}`	or
`\@`⟨*type*⟩`{`⟨*nickname*⟩`}{`⟨*external name*⟩`}`	for a scaled font.

where a ⟨*type*⟩ is one of the control sequences `\f`, `\m` or `\M`, and ⟨*nickname*⟩ is the two- or three-letter nickname used for the font (without any leading backslash), for example `rm`, `it`, `bf`. There must be a font fam called `\`⟨*nickname*⟩`fam` (`\rmfam`, `\itfam` etc. are already defined).

If there is no `\@`, then the ⟨*partial name*⟩ is a font name sans the size specification, such as `cmr`. The size in points will be appended to this (`cmr` + 10 pt = `cmr10`). If `\@` is included then the ⟨*external name*⟩ is the complete font name, such as `cmr10` or `ptmr`. This will be followed by '`␣at␣10pt␣`', say.

The ⟨*type*⟩ specifies how much support for mathematics this font requires. This is because maths mode requires that all fonts that might be used in a formula be loaded (because the font tables `\textfont`, `\scriptfont` and `\scriptscriptfont` must be set). The code `\f` means that the font is not used in maths, `\m` means that `\textfont` and `\scriptfont` will be set correctly for this fam, and `\M` means that `\scriptscriptfont` will also be set. In these cases there must be a corresponding token ending in '`-fam`' that expands to the fam number.

The fontset name is a sequence of ⟨*letter*⟩s, like '`body`', '`note`', '`script`', '`heading`'. This is turned into a fontset selection macro by adding '`\`' to the front and '`fonts`' to the end (e.g., `\bodyfonts`). The csnames for loaded fonts are formed from the fontset name + the nickname (e.g., `\bodyrm`).

5.1. Defining the csnames for a fontset

The macro `\xfontset` defines all the csnames for the fonts in a fontset, without defining a '`-fonts`' macro. This is used to define a fontset that is never selected in its own right (e.g., its fonts are used only as subscripts and superscripts), and also used internally by the `\fontset` command. Its arguments are `#1` the fontset name, `#2` the csname of a template macro, and `#3` a ⟨*number*⟩ that specifies the font size (sans the '`pt`').

```
\def\xfontset#1#2#3{
    \def\f##1##2{\expcs\loadfont{#1##1}{##2#3}}
    \let\m=\f \let\M=\f
    \def\@##1##2##3{\expcs\loadfont{#1##2}{##3 at #3pt }}
    #2
}
```

5.2. Defining a complete fontset

The user command `\fontset` is used to define a complete fontset. Its parameters are `#1` (a string of letters) is the fontset name, `#2` (a csname) is a template macro, `#3` (a ⟨*number*⟩) is size in points, `#4` (a ⟨*skip*⟩) is baseline skip, `#5` (a fontset name) is the scriptstyle fontset, and `#6` (a fontset name) is the scriptscriptstyle fontset.

```
% Set up a fontset -- define \#1fonts
\def\fontset#1#2#3#4#5#6{%
    \xfontset{#1}{#2}{#3}%
```

Now to define the `\#1fonts` macro. When demand-loading, this macro will call `\#1mathfonts` (to ensure the fonts needed for maths are loaded). Then it will call `\FSELnicknames` to define `\rm`, `\it`, etc., and to define `\textfont\rmfam`, etc. Finally it will set the baseline skip and related parameters and switch to the new `\rm` font.

```
    \expcs\edef{#1fonts}{%
       \ifFSELautoload \noexpcs{#1mathsfonts}\fi
       \noexpand\FSELnicknames{#1}{#5}{#6}\noexpand#2%
       \noexpand\setbaselineskip{#4}%
       \noexpand\rm
    }%
```

If we are demand-loading, we must define `\#1mathsfonts` as well.

```
    \ifFSELautoload
       \expcs\def{#1mathsfonts}{\FSELloadmaths{#1}{#5}{#6}#2}%
    \fi
}
```

5.3. Setting font nicknames

`\FSELnicknames` gives definitions to `\f`, `\m` and `\M` so that expanding the template macro defines `\rm` and the like. If the fontset name is `ffff`, and a nickname `xx` is introduced with `\f`, the macro `\xx` is defined to `\ffffxx`. If it is introduced with `\m`, then the font table entries `\textfont\xxfam` and `\scriptfont\xxfam` are also set, and `\xx` expands to '`\fam\xxfam \ffffxx`'. If it is introduced with `\M` then `\scriptscriptfont\xxfam` is also set.

The parameters are #1 the fontset name, #2 the fontset name for scriptstyle, and #3 the fontset name for scriptscriptstyle. The implicit fourth parameter is the template macro.

```
 71 \def\FSELnicknames#1#2#3{%
 72     \let\@\relax
 73     \def\f##1##2{%
 74        \expcs\edef{##1}{\noexpcs{#1##1}}%
 75     }%
 76     \def\m##1##2{%
 77        \expcs\textfont{##1fam}\csname#1##1\endcsname
 78        \expcs\scriptfont{##1fam}\csname#2##1\endcsname
 79        \expcs\edef{##1}{%
 80            \fam\expcs\noexpand{##1fam}%
 81            \expcs\noexpand{#1##1}%
 82        }%
 83     }%
 84     \def\M##1{%
 85        \expcs\scriptscriptfont{##1fam}\csname#3##1\endcsname
 86        \m{##1}%
 87     }%
 88 }
```

5.4. *Setting the baseline skip*

The second helper macro, `\setbaselineskip`, sets `\baselineskip` and a bunch of related paramaters and pseudo-parameters like `\smallskipamount`. It takes one parameter, a ⟨*skip*⟩.

Note *My definitions for the skips that go before and after displays put less white-space around displays than is set in plain TEX. This can be changed by redefining this macro in a style file.*

```
 89 \def\setbaselineskip#1{%
 90     \baselineskip#1\relax \normalbaselineskip\baselineskip
 91     \jot0.25\baselineskip
 92     \smallskipamount 0.25\baselineskip plus 0.083\baselineskip
 93        minus 0.083\baselineskip
 94     \medskipamount 0.5\baselineskip plus 0.167\baselineskip
 95        minus 0.167\baselineskip
 96     \bigskipamount 1\baselineskip plus 0.333\baselineskip
 97        minus 0.333\baselineskip
 98     \abovedisplayskip\medskipamount
 99     \abovedisplayshortskip\abovedisplayskip
100     \advance\abovedisplayshortskip-1\abovedisplayskip
101     \belowdisplayskip\medskipamount
102     \belowdisplayshortskip\smallskipamount
103 }
```

5.5. *Loading maths fonts*

Finally, we need to force the fonts used in maths at a given size to be loaded. for `\f` fonts this does nothing; for `\m` fonts it loads the text and script fonts; for `\M` fonts it also loads the scriptscript font.

This is done the first time this fontset is selected, even if no formulas are used (rather than trying to do something complicated like use `\everymath`...). Because this macro only needs to be used once for each fontset, it finishes by redefining `\#1mathfonts` to be the same as `\relax`.

Its parameters are #1 the fontset name, #2 the fontset name for scriptstyle, and #3 the fontset name for scriptscriptstyle.

```
104 \def\FSELloadmaths#1#2#3{%
105     \let\@\relax \def\f##1##2{}%
106     \def\m##1##2{\csname#1##1\endcsname \csname#2##1\endcsname}%
107     \def\M##1{\csname#3##1\endcsname \m{##1}}%
108     \global\expcs\let{#1mathsfonts}\relax
109 }
```

6. Finishing up

We define `\rmfam`, `\mifam` and `\syfam` as aliases for the numbers 0, 1 and 2. This is so that the maths fonts may be included in the font templates without any special arrangements. The names `\itfam`, `\bffam`, `\ttfam`, and `\slfam` are set in `plain.tex`.

```
110 \chardef\rmfam=0 \chardef\mifam=1 \chardef\syfam=2
```

Note that we do not include an alias for the the maths extension font's fam number. This is because there is only one maths extension font—`cmex10`—used for all sizes, so it does not belong in any fontset.

7. Summary of user commands

The following lists the user commands provided by FSEL. A ⟨*fontset name*⟩ is a sequence of letters like `note`; ⟨*points*⟩ is a TEX ⟨*number*⟩ representing size in points (without any final `pt`); an ⟨*external name*⟩ is the external name for a font (e.g., '`cmr12`').

```
\autoloadfonts
\loadfont⟨csname⟩{⟨external name⟩}
\xfontset{⟨fontset name⟩}⟨csname⟩{⟨points⟩}
\fontset{⟨fontset name⟩}⟨csname⟩{⟨points⟩}{⟨skip⟩}{⟨fontset name⟩}{⟨fontset name⟩}
\def\everyloadfont#1#2{ ... }
\def\setbaselineskip#1{ ... }
```

8. Bugs

FSEL clobbers the macros `\f`, `\m`, `\M` and `\@`.