styledef.tex

styledef.tex: Macros to selectively input parts of a file.
Version: 1.0 (16 April 1991) (doc: 19 Sep 1991)

Consider the following situation: You write a set of TeX macros to typeset letters. These macros must set fonts, margins, offsets, etc. However, you want to use different styles for different types of letters. For instance, a short letter may need extra white space at the top to get a full page, whereas this is not needed for a letter several pages long. And a business letter has a standard header, with subject and references, whereas a small note to a friend only has a date. Probably the macros for the different styles have a lot in common, a reason to put them all in one file. But the different styles may have some conflicting definitions, of which you only want to include one. Styledef.tex* provides some macros to `\input` from a file only parts belonging to a particular "style." (Partial inclusion can also be useful for other things than styles; see the end of this manual section for an example.)

Before we explain the macros from styledef.tex, let us look at two other solutions. First, say the only difference between styles "wide" and "narrow" is the width of the text. Then we can write macros `\widestyle` and `\narrowstyle` that set the appropriate width. As long as the differences between styles are small and easily parametrized, this method works fine. If there are many differences, however, the macros for setting the style soon get very large. Furthermore, it is hard too share parts of styles.

A second method is the one used by LaTeX. One makes one file "letter.sty" with all the common definitions for letters; an additional file is made for each style. LaTeX's `\documentstyle` macro is then used to include the proper files. E.g., one could write

```
\documentstyle[business,narrow]{letter}
```

to write a letter with the "business" and "narrow" style options. This method allows sharing of common definitions, and also allows the combination of different style options. The disadvantage is that it soon leads to a very large number of files, many of them with only a few definitions.

With the macros in styledef.tex, you would use only one or two files. Typical would be to have one file, "letter.tex," with all the common definitions for letters, and one file, "letter.style," with all the different styles. Only some parts of the latter file are included, through the macros below. Alternatively, everything can be put in a single file. (Using two files has the advantage that "letter.tex" can be shared by several users, who each have there own personalized "letter.style.") Styledef.tex is of course especially useful with plain TeX, since that does not have an equivalent mechanism. However, the macros can also be used with LaTeX; later on we describe how to make the `\documentstyle` command interact with the styledef macros.

First we describe how to `\input` only parts of a file. In order to use any of these macros, you first have to `\input styledef`. Then you have to allocate one or more "style registers": a style register called "`\myst`" is allocated with the command

```
\newstyle\myst
```

You can give a new value to a style register through the `\setstyle` command. It takes a style register and a list of styles as arguments. E.g.,

```
\setstyle\myst{business, narrow}
```

For this command, as for all others, a style list is a list of style names separated by spaces and/or commas; the whole list must be enclosed in braces. The style names themselves can be any names that are meaningful to the macro package you are using. The `\addstyle` command is used in the same way as `\setstyle`, but it adds one or more styles to the style register without removing the old contents. You can use `\the\myst` to get the contents of the style register, and style registers can be assigned to each other using '`=`'. The last operation involving style registers is `\readstyle`. It takes a style register and a file name: The command

```
\readstyle\myst{letter.style}
```

* You also need Midnight/dolines.tex and Midnight/loop.tex, which will be `\input` automatically.

will `\input` file letter.style, but includes only those parts of it that belong to one of the styles in register `\myst`. That is all that is needed to selectively `\input` a file. The other macros from styledef.tex and the remainder of this manual section describe how to indicate in a file like letter.style which parts belong to which styles. If you don't intend to create such files yourself, there is no need to read on.

There are only three commands that govern which parts of a file belong to a particular style: `\styledef`, `\negstyledef`, and `\endstyle`. The first two are followed by a brace-enclosed list of styles (but not by a style register). When a file is read with `\readstyle`, the file is just `\input` by TeX. However, when in that file a command

```
\styledef{business, long}
```

is encountered, TeX checks whether the style register that is used with `\readstyle` contains style "business" or style "long." If at least one of them is contained in the register, TeX continues to input the file. But if none of the styles match, TeX skips lines of the file until an `\endstyle` command is encountered. `\negstyledef` has exactly the opposite effect: if any style matches, the file is skipped till the next `\endstyle` command; only if no style matches is this part of the file included. The `\endstyle` command must occur on a line by itself, with nothing, not even a comment, following it. The `\endstyle` command stops TeX from skipping text; if it is encountered when TeX is not skipping text, it is equivalent to `\relax`.

Text that is not enclosed by a `\styledef` ...`\endstyle` (or `\negstyledef` equivalent) is always included. `\styledef` commands can be used within each other's range, but they do not actually nest: when part of the text is skipped, skipping always stops at the first `\endstyle`; any intervening `\styledef` commands are also skipped. E.g.,

```
\styledef{business}
\negstyledef{narrow}
 ...
\endstyle
```

The text corresponding to '...' (which does not include any `\endstyle`) is only included if the style register contains style "business" and does not contain "narrow." If style "business" is not contained in the register, the `\negstyledef` command as well as the '...' are skipped. As a catch-all you may want to put an `\endstyle` at the end of the included file: if you missed an `\endstyle` somewhere, this catch-all will at least restrict the effects to this file.

The `\readstyle` command uses an internal style register to store which styles must be included. Therefore, if a file which is being read with `\readstyle`, uses `\readstyle` to read yet another file, the original list of styles will be lost. Although this situation seldomly occurs, there is a command `\savereadstyle` which takes a style register as argument and saves the internal style list in this register. Of course, if the second file should be read with exactly the same style list, you should just use `\input` instead of `\readstyle`. To restore the original style list, you use `\setreadstyle`, which also takes a style register as argument. `\setreadstyle` can also be used instead of `\readstyle`, if you want to select certain parts of the current file rather than `\input` a different file.

Style names can be built from any characters, except for spaces, commas, dots, and some characters with a special meaning to TeX (e.g., braces, hash signs (`#`), and percentage signs). Also, characters in a style name must have the same catcode in the `\styledef` command as in the `\setstyle` command. When a `\styledef` or `\negstyledef` command is encountered, the following conditions must hold: The only escape character (catcode 0) is '`\`'; the only grouping characters (catcodes 1 and 2) are '`{`' and '`}`'; the only parameter character (catcode 6) is '`#`'; and the only comment character (catcode 14) is '`%`'. After the `\styledef` command these catcodes may be changed, as long as the conditions hold again when the next `\styledef` is encountered. These particular catcode assignments are so standard that this rule is unlikely to ever cause problems.

Suppose you have written a file letter.tex, which is to be used by several people. You also have a file letter.style with some standard styles; people can copy and change this file to suit their own needs. To make life easier for those who only want to use your standard styles, you may not want to force them to learn about styledef.tex. Instead, you can define a macro `\style` which always reads a file with name letter.style. Its usage would be:

```
\input letter
\style business, long, narrow.
 ...
```

so that users need not be concerned about style registers etc. Here is part of an appropriate letter.tex:

```
\input styledef
\newstyle\letterstyle

\def\style#1.% terminate the argument with a dot
   {\setstyle\letterstyle{#1}
    \readstyle\letterstyle{letter.style}
   }
```

Something similar can be done with LaTeX's `\documentstyle`, provided you know which styles can appear in letter.style. This is done as follows: first, rename letter.tex to letter.sty, so that LaTeX can read it. Then add the following to the end of that file:

```
\newstyle\lst
\def\ds@business{\addstyle\lst{business}}
\def\ds@narrow{\addstyle\lst{narrow}}
 ... % same for every style
\@options
\readstyle\lst{letter.style}
```

You can then write `\documentstyle[narrow]{letter}`. LaTeX takes this as a command to read letter.sty. The `\@options` command causes LaTeX to execute, for every option 'xyz,' the command `\ds@xyz`, if it exists. All options for which this command does not exist lead to the input of a file xyz.sty, so that standard LaTeX options can still be used. Note that letter.sty is here the main style file (only there can `\@options` be used); be warned that it is not easy to write a main style file that satisfies all of LaTeX's assumptions.

Although we have suggested that these macros are suitable to handle different "styles," selective inclusion can also be useful in contexts unrelated to styles. For instance, you may write a paper which has some details and speculations which are not relevant for every reader. You can write the paper like this:

```
 ... % relevant thoughts
\styledef{details}
 ... % messy details
\endstyle
\negstyledef{details}
Details are left as an exercise for the reader.
\endstyle
 ... % important stuff
\styledef{speculations}
 ... % wild fantasies
\endstyle
```

If you want to print a version with speculations but without details, you can start the file with

```
\newstyle\print
\setstyle\print{speculations}
\setreadstyle\print
 ...
```