

A style option to create nice frames with the **dingbat** fonts*

Marcus Ohlhaut
Westendstraße 231
D-80686 München
marcus_ohlhaut@m4.maus.de

Printed October 23, 1996

Abstract

This article describes a new style option that can be used with the document styles that are distributed with the L^AT_EX distributions. It defines new commands to frame material with decorative frames, using **dingbat** fonts.

Contents

1	Introduction	1
2	The user interface	2
3	The implementation	2
3.1	The DOCSTRIP modules	2
3.2	Producing the documentation	2
3.3	The example	3
3.4	Useful fonts	4
3.5	The code	4

1 Introduction

When I set out to become familiar with all the fonts available for T_EX, I came across the **dingbat** font. Only a few glyphs have been defined in that font, and some of them are intended to yield nice frames when set appropriately.

I didn't want to work out the details of arranging the glyphs each time I needed to frame text, and hence wrote this little package to do the work for me.

When testing this, I noticed that one of the glyphs was wrongly coded in the original **dingbat** distribution. Together with this package, a corrected version of **dingbat** is distributed.

This package has no options. It requires the package `calc` by Kresten K. Thorup and Frank Jensen.

*This file has version number 1.1b – last revision 1996/10/20.

2 The user interface

This package defines three commands, `\niceframe`, `\curlyframe` and `\artdecoframe`. All three take one mandatory and one optional argument. The mandatory argument specifies the material that is to be framed (anything which can go into a `\vbox`), whereas the optional argument specifies the final width of the frame.

If the optional argument is omitted, the frame will be exactly of width `\textwidth`. The height of the frame is calculated from the natural height of the enclosed material. These frames have to be set into their own paragraphs, otherwise the width of the surrounding text will not be correct.

The package also defines a more general macro called `\generalframe` to allow control over the symbols which are used to build the frame.

3 The implementation

3.1 The DOCSTRIP modules

The following modules are used in the implementation to direct DOCSTRIP in generating the external files:

driver	produce the documentation driver file
package	produce the package file
example	produce an example file

3.2 Producing the documentation

A short driver is provided that can be extracted (if necessary) by the DOCSTRIP program provided with $\text{\LaTeX} 2_{\epsilon}$.

```
1 \(*driver)
2 \documentclass{ltxdoc}
3
4 % dimensions from ltugboat.sty:
5 \setlength\textwidth{31pc} \setlength\textheight{54pc}
6 \setlength{\parindent}{0pt}
7 \setlength{\parskip}{2pt plus 1pt minus 1pt}
8 \setlength{\oddsidemargin}{8pc}
9 \setlength{\marginparwidth}{8pc}
10 \setlength{\topmargin}{-2.5pc}
11 \setlength{\headsep}{20pt}
12 \setlength{\columnsep}{1.5pc}
13 \setlength{\columnwidth}{18.75pc}
14 \EnableCrossrefs
15 \RecordChanges
16 \CodelineIndex
17
18 \MakeShortVerb{\|}
19 \begin{document}
20
21 \DocInput{niceframe.dtx}
22
23 \end{document}
24 \(/driver)
```

3.3 The example

This is just a short L^AT_EX-file to test the installation. Install the `niceframe` package, and run `latex example.tex`.

```
25 (*example)
26 \documentclass[a4paper]{article}
27 \usepackage{niceframe}
28
29 \begin{document}
30 \pagestyle{empty}
31
32 \parindent 0pt
```

First, a `\niceframe` that spans the whole `\textwidth`.

```
33 \niceframe{%
34   \begin{center}
35     May the road rise to meet you\\
36     May the wind be always at your back\\
37     May the sun shine warm upon your face\\
38     The rain fall soft upon your fields\\
39     And until we meet again\\
40     May God hold you in the hollow of his hand\\
41   \end{center}
42   \bigskip
43   \hfill The Blessing of St.~Patrick
44 }
45 \vfill
```

Then, a centered `\curlyframe` that is only `0.75\textwidth` wide.

```
46 \centerline{%
47   \curlyframe[0.75\textwidth]{%
48     \begin{center}
49       \LARGE I am not flexible!\\
50       Just disorganized!\\
51     \end{center}
52 }}
53 \vfill
```

And an `\artdecoframe` for the philosophical bit.

```
54 \centerline{or is it\dots}
55 \vfill
56 \artdecoframe{%
57   \begin{center}
58     \LARGE I am not disorganized!\\
59     Just flexible!\\
60   \end{center}
61 }
62 \newpage
```

The following three frames are examples of `\generalframe`.

```
63 \font\border=karta15
64 \generalframe{\border\char'307}{\border\char'324}{\border\char'322}
65             {\border\char'310}             {\border\char'323}
66             {\border\char'174}{\border\char'325}{\border\char'175}
67             {This is a \texttt{generalframe}, using eight different
68             characters from font \texttt{karta15} as parts of the
69             framework. One might use it to point other people to
70             important notes, hints and/or messages.
71             }
```

```

72 \vfill
73 \font\border=umranda
74 \generalframe{\border\char'136}{\border\char'137}{\border\char'140}
75         {\border\char'145}                {\border\char'141}
76         {\border\char'144}{\border\char'143}{\border\char'142}
77         {This \texttt{generalframe} uses symbols from Alexander
78         Schrell's \texttt{umranda} (his eMail is:
79         alexander\_ schrell@w2.maus.de) collection.
80         The material to be framed consists of quite a few lines to
81         emphasize the fact that virtually anything can go
82         inside a \texttt{generalframe}. Note that we are limited
83         by \TeX's memory as the material to frame has to be passed
84         as an argument. You can't cheat by passing a \texttt{vbox}
85         that has already been set (as you might with some other
86         macros) since \texttt{generalframe} has to re-set all of it.
87         }
88 \vfill
89 \font\border=umrandb
90 \generalframe{\border\char'165}{\border\char'151}{\border\char'164}%
91         {\border\char'150}                {\border\char'150}%
92         {\border\char'166}{\border\char'151}{\border\char'167}
93         {Last, but not least! This \texttt{generalframe} (with
94         symbols from \texttt{umrandb}) displays some math:
95         $$ E = mc^2 $$
96         which everyone will immediately recognize as the famous
97         Einstein formula for energy-mass-equivalence.
98         }
99 \end{document}
100 /example)

```

3.4 Useful fonts

The font used for setting the frames is the **dingbat** font by Doug Henderson. Since the original distribution contained a wrongly coded char, I have included a corrected version with this package. Except from the code for `char("e")`, this is Doug's original file.

I have also added the fonts **bbding10**, **karta15**, **umranda** and **umrandb** for they are used in the examples.

3.5 The code

We begin by identifying the version of this file on the terminal and in the transcript file and by loading the package `calc`.

```

101 <*package>
102 \NeedsTeXFormat{LaTeX2e}[1994/06/01]
103 \ProvidesPackage{niceframe}[\filedate\space v\fileversion\space niceframe package (MO)]
104 \typeout{Package: niceframe v\fileversion\space <\filedate> (Marcus Ohlhaut)}
105 \RequirePackage{calc}

```

We then load the font used for setting the standard frames.

```

106 \font\ding dingbat scaled 1200

```

A few internal dimensions need to be defined and initialized.

```

107 \newlength{\nicefr@mechar}
108 \settowidth{\nicefr@mechar}{\ding\char'141}
109 \newlength{\nicefr@mewidth}
110 \setlength{\nicefr@mewidth}{\hspace}

```

```

111 \newlength{\nicefr@meheight}
112 \setlength{\nicefr@meheight}{\vsize}
113 \newlength{\@ldhsize}
114 \setlength{\@ldhsize}{\hsize}

```

`\upd@ublerulefill` These internal macros define a set of leaders for each of the four sides of `\niceframe`.

`\dnd@ublerulefill` The appropriate chars are taken from the **dingbat** font.

```

\dld@ublerulefill 115 \newcommand{\upd@ublerulefill}{\xleaders\hbox to 10pt
\dnd@ublerulefill 116 {\hss\ding\char'142 \hss}\hfill}
117 \newcommand{\dnd@ublerulefill}{\xleaders\hbox to 10pt
118 {\hss\ding\char'147 \hss}\hfill}
119 \newcommand{\ltd@ublerulefill}{\xleaders\vbox to 10pt
120 {\vss\hbox{\ding\char'144}\vss}\vfill}
\drt@ublerulefill 121 \newcommand{\rtd@ublerulefill}{\xleaders\vbox to 10pt
122 {\vss\hbox{\ding\char'145}\vss}\vfill}

```

`\niceframe` Finally, we define the user interfaces.

```

123 \newcommand{\niceframe}[2][\textwidth]{

```

First, we have to set the material that is passed to `\niceframe` to an appropriate width. That width is the requested width of the final `\vbox` minus twice the size of the chars that make up the corners of the frame (ie. `\nicefr@mechar`). The set material is stored for further use in `\box0`.

```

124 \setlength{\hsize}{#1 - 2\nicefr@mechar}
125 \setbox0=\vbox{#2}

```

We then use height and depth of `\box0` to calculate the required height of the `\vbox` that encloses the vertical leaders.

```

126 \setlength{\nicefr@meheight}{\ht0 + \dp0}

```

Similarly, we then use the width of `\box0` for enclosing the horizontal leaders.

```

127 \setlength{\nicefr@mewidth}{\wd0 + 2\nicefr@mechar}

```

Finally, when all the relevant dimensions have been determined, the frame is compiled. We start with the top row (corner char, horizontal leader, corner char), set the middle part (vertical leader, things to frame, vertical leader) and finish with the bottom row (corner char, horizontal leader, corner char).

```

128 \vbox{
129 \hbox to\nicefr@mewidth{\ding\char'141\upd@ublerulefill\char'143}
130 \hbox to\nicefr@mewidth{\vbox to\nicefr@meheight{\ltd@ublerulefill}
131 \hss\raise\dp0\box0\hss
132 \vbox to\nicefr@meheight{\rtd@ublerulefill}}
133 \hbox to\nicefr@mewidth{\ding\char'146\dnd@ublerulefill\char'150}
134 }
135 }}

```

`\curlyframe` This macro works exactly like `\niceframe`. The only difference is that we use stretchable space instead of leaders and different glyphs for the four corners (obviously).

We also have to move the text a bit closer to the corners due to their shape.

```

136 \newcommand{\curlyframe}[2][\textwidth]{
137 \setlength{\hsize}{#1 - 2\nicefr@mechar}
138 \setbox0=\vbox{#2}
139 \setlength{\nicefr@meheight}{\ht0 + \dp0}
140 \setlength{\nicefr@mewidth}{\wd0 + 2\nicefr@mechar}
141 \vbox{
142 \hbox to\nicefr@mewidth{\ding\char'105\hfill\char'106}

```

```

143 \vskip-\baselineskip
144 \hbox to\nicefr@mewidth{\hss\raise\dp0\box0\hss}
145 \vskip-\baselineskip
146 \hbox to\nicefr@mewidth{\ding\char'110\hfill\char'107}
147 }
148 }}

```

`\artdecoframe` This is just a variation of `\curlyframe`.

```

149 \newcommand{\artdecoframe}[2][\textwidth]{
150 \setlength{\hsize}{#1 - 2\nicefr@mechar}
151 \setbox0=\vbox{#2}
152 \setlength{\nicefr@meheight}{\ht0 + \dp0}
153 \setlength{\nicefr@mewidth}{\wd0 + 2\nicefr@mechar}
154 \vbox{
155 \hbox to\nicefr@mewidth{\ding\char'115\hfill\char'114}
156 \hbox to\nicefr@mewidth{\hss\raise\dp0\box0\hss}
157 \hbox to\nicefr@mewidth{\ding\char'112\hfill\char'113}
158 }
159 }}

```

`\generalframe` Finally, I have designed a more general macro for framing. `\generalframe` takes nine parameters (right to the limit) and produces a `\vbox`. The width of this box is less than (or equal to) the current `\hsize`; and it consists of a border generated from the characters in the first eight arguments and a box which contains the material passed in the very last argument.

The first eight parameters to `\generalframe` are used as follows:

- #1 is the character for the top left corner,
- #2 builds the top horizontal leader,
- #3 is the character for the top right corner,
- #4 builds the left vertical leader,
- #5 builds the right vertical leader,
- #6 is the character for the bottom left corner,
- #7 builds the top horizontal leader,
- #8 is the character for the bottom right corner,

This box is centered with respect to the frame, and there is at least `\fboxsep` between the frame and the enclosed material. To ensure that the inner box is calculated properly, all the characters for framing need to be of equal width and height.

We first define some internal quantities. `times` holds the number `\hsize div \fr@mecharTT` which is required in the process of rounding, `\fr@mecharTT` and `\fr@mecharLL` hold the width and height of the characters that make up the top/bottom and left/right parts of the frame, respectively.

`\fr@mewidth` holds the final width of the frame and is always less than or equal to the current `\hsize`, whereas `\fr@meheight` holds the final height of the frame minus the heights of the top and bottom row.

```

160 \newcounter{times}
161 \newlength{\fr@mecharTT}
162 \newlength{\fr@mecharLL}
163 \newlength{\fr@mewidth}
164 \newlength{\fr@meheight}

```

`\generalframe` initializes `\fr@mecharTT` and `\fr@mecharLL` every time it is called, relying on the fact that the user specifies the eight border characters in such a way that #1, #2, #3, #6, #7 and #8 have the same width and #1, #3, #4, #5, #6 and #8 have the same height.

```
165 \newcommand{\generalframe}[9]{
166 \settowidth{\fr@mecharTT}{#2}
167 \settoheight{\fr@mecharLL}{#4}
```

In order to enable a packed alignment of the horizontal leaders, we have to round `\fr@mewidth` to an integer multiple of `\fr@mecharTT`.

```
168 \setcounter{times}{1 * \ratio{\hsize}{\fr@mecharTT}}
169 \setlength{\fr@mewidth}{\fr@mecharTT * \value{times}}
```

`\fr@mewidth` is then used to calculate the `\hsize` of the innermost box, allowing for some horizontal space at either side. The box is set and stored in `\box0` for later use.

```
170 \setlength{\hsize}{\fr@mewidth - 2\fr@mecharTT - 2\fbboxsep}
171 \setbox0=\vbox{#9}
```

We then calculate `\fr@meheight` (height plus depth of `\box0`, with some vertical space added) and round it to an integer multiple of `\fr@mecharLL`. We add one to ensure at least one character in the vertical leaders.

```
172 \setlength{\fr@meheight}{\ht0 + \dp0 + 2\fbboxsep}
173 \setcounter{times}{1 * \ratio{\fr@meheight}{\fr@mecharLL}}
174 \setcounter{times}{\value{times} + 1}
175 \setlength{\fr@meheight}{\fr@mecharLL * \value{times}}
```

Four leaders are used to construct the main part of the frame.

```
176 \newcommand{\up@fill}{\leaders\hbox{#2}\hfill}
177 \newcommand{\lt@fill}{\leaders\vbox{\hbox{#4}}\vfill}
178 \newcommand{\rt@fill}{\leaders\vbox{\hbox{#5}}\vfill}
179 \newcommand{\dn@fill}{\leaders\hbox{#7}\hfill}
```

Finally, everything is packed together, centered nicely.

```
180 \vbox{%
181 \hbox to\fr@mewidth{#1\up@fill#3}\nointerlineskip
182 \hbox to\fr@mewidth{\vbox to\fr@meheight{\lt@fill}%
183 \hfill%
184 \vbox to\fr@meheight{\vfill\box0\vfill}%
185 \hfill%
186 \vbox to\fr@meheight{\rt@fill}%
187 }\nointerlineskip
188 \hbox to\fr@mewidth{#6\dn@fill#8}\nointerlineskip
189 }
190 }}
```

That's all there is.

```
191 \endpackage)
```

Index

Numbers written in *italic* refer to the page where the corresponding entry is described, the ones underlined to the code line of the definition, the rest to the code lines where the entry is used.

`artdecoframe=` `\subitem *+artdecoframe 56m 149, 63m-66, 7357it89-92curlyframe+,`

